

A new straightening algorithm of bracket algebra

Changpeng Shao, Hongbo Li

KLMM, Academy of Mathematics and Systems Science
Chinese Academy of Sciences
Beijing 100190, China
shaochangpeng11@mails.ucas.ac.cn, hli@mmrc.iss.ac.cn

Abstract. Straightening algorithm is always a main task in bracket algebra. And this is a very complicate procedure. Until now, some beautiful algorithms have been provided. White's implementation algorithms [17] seem the most efficient one. This paper is devoted to give another new algorithm as a byproduct of invariant division put forward in [9], which seems more suitable to straight high degree tableaux. The examples presented in section 5 show that this new algorithm is more efficient than White's algorithm when the degree of tableaux is ≥ 4 .

Key words: Bracket algebra; Capelli operator; invariant division; straightening algorithm.

1 Introduction

The First Fundamental Theorem of Classical Invariant Theory [6], [16] states that all the invariants of the general linear group acting on a vector space are homogeneous polynomials of brackets (i.e., determinant of n columns of vectors of dimension n). Bracket algebra is the polynomial ring with brackets as its variables. Thus it is a quotient ring, due of the existence of algebraic dependence among determinants. Then it would be very helpful to study the normal forms of polynomials in the bracket algebra, because of its uniqueness. A classical theorem [8], [14] says that all normal bracket polynomials are a linear combination of *straight* (or *standard*) bracket monomials (i.e., in the tableau form, each row is ascending, and each column is non-descending under the lexicographical order of vectors). As we know, the way of changing any polynomial into the normal form can be achieved by Gröbner basis method. The ideal generated by the algebraic dependence among determinants has a Gröbner basis composed of *van der Waerden polynomials*. And the way of changing a bracket polynomial into its normal form is called *straightening*.

Until now, there are at least four different kinds of versions straightening algorithms: the van der Waerden straightening algorithm (i.e., classical straightening algorithm) [8], [14], N. White's implementation on the classical straightening algorithm [17], Rota's straightening algorithm [4] and the dotted straightening algorithm [10]. These methods have their own advantages and disadvantages.

The van der Waerden straightening algorithm is a procedure of reduction just like Gröbner basis. Such a normal form procedure is the *straightening law* due to A. Young [20]. This algorithm find the normal form directly without any other auxiliaries. While in the process, there will be a lot variations when related to high degree's bracket monomials, which will reduce the efficiency of this straightening algorithm.

N. White made some analysis on the van der Waerden syzygies and the multiple syzygies. And he gave five different implementation straightening algorithms, and shown that under some cases using the multiple syzygies is more efficient, while under some other cases using the van der Waerden syzygies is more efficient. As shown by examples, N. White's implementation algorithms are much better than the classical straightening algorithm.

G. C. Rota adopted the advantages of Capelli operator, and changed the straightening process into the process of solving a down-triangle system of linear equations. This algorithm has been successfully used to compute the straightening coefficients [3], [5], [7]. An apparent defect of Rota's straightening algorithm in practical is that it needed to find all the straightening tableaux with the same content of the given tableau.

The dotted straightening algorithm is a straightening algorithm about dotted bracket expressions. Under some conditions, an ordinary bracket polynomial can be written as a dotted bracket expressions. This is a much more efficient straightening algorithm than the van der Waerden straightening algorithm and multiple syzygies straightening algorithm for bracket polynomials that have a dotted form. The dotted straightening algorithm can be used to achieve multilinear Cayley factorization [14], [19].

In this paper, based on van der Waerden straightening algorithm and invariant division theory over bracket algebra [9], we introduce another new straightening algorithm in bracket algebra. As shown by examples in section 5, the efficiency of our algorithm and N. White's algorithm are almost equally matched in the case of 3×3 , but in the $m \times 3$ ($m \geq 4$) case, our algorithm seems much better than N. White's algorithm.

This new straightening algorithm rely on a new concept called column bracket. The relationship between column brackets, the brackets and the coordinate monomials can be descried as "coordinate monomials $<$ column brackets $<$ brackets". This is because bracket can be expressed as polynomials of column brackets, and column bracket is defined over coordinate monomials. Column bracket has the same form of tableaux, while contains more symmetries than brackets. Moreover, column bracket can be viewed as another version of Capelli operator, because they all reflect the sign of substitutions in the straightening procedure.

This paper is organized as follows. In section 2, we will review some necessary results from the paper [9]. In section 3, we will briefly introduce three straightening algorithms. Section 4 is the main parts of this paper, it is devoted to give the definition of column bracket, and show our new straightening algorithm. In

section 5, we will make an analysis on the efficiency of the the three famous straightening algorithm and our straightening algorithm via examples.

Notation: in this paper, all the bold letters such as $\mathbf{b}_i, \mathbf{b}_{ij}, \mathbf{c}_i, \mathbf{d}_i, \dots$ without other description refers to elements of $\{\mathbf{a}_1, \dots, \mathbf{a}_m\}$.

2 Brief introduction of bracket algebra

In this section, we will briefly recall some results from the paper [9] for the following requirements. As for the theory of bracket algebra, we refer to [8] and [14].

Let $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m$ be m vectors in a n -dimensional \mathbb{K} -vector space \mathcal{V}^n ($m > n$). The n -dimensional bracket algebra generated by these vectors is denoted as $\mathcal{B}[\{\mathbf{a}_i : 1 \leq i \leq m\}]$, or just as $\mathcal{B}[\{\mathbf{a}_i\}]$. Taking the lexicographical order among the m vectors $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m$:

$$\mathbf{a}_1 \prec \mathbf{a}_2 \prec \dots \prec \mathbf{a}_m. \quad (2.1)$$

Definition 1. Let $T = \begin{bmatrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \dots & \mathbf{b}_{1s} \\ \mathbf{b}_{21} & \mathbf{b}_{22} & \dots & \mathbf{b}_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{b}_{r1} & \mathbf{b}_{r2} & \dots & \mathbf{b}_{rs} \end{bmatrix}$ be a bracket monomial, then define

$$\text{vec}(T) := (\mathbf{b}_{11}, \mathbf{b}_{21}, \dots, \mathbf{b}_{r1}, \mathbf{b}_{12}, \mathbf{b}_{22}, \dots, \mathbf{b}_{r2}, \dots, \mathbf{b}_{1s}, \mathbf{b}_{2s}, \dots, \mathbf{b}_{rs}). \quad (2.2)$$

Definition 2. (negative column order) Let f, g be two straight bracket monomials, with degree d_f, d_g respectively, then we say that $f \succ g$ under the negative column order if

- (1). $d_f > d_g$; or
- (2). $d_f = d_g$, and $\text{vec}(f) \prec \text{vec}(g)$ in the lexicographical order.

Example 1. $f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}, g = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}$, then $\text{vec}(f) = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_5, \mathbf{a}_6)$, $\text{vec}(g) = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_4, \mathbf{a}_6)$. Since $\text{vec}(f) \prec \text{vec}(g)$ lexicographically, so $f \succ g$ in the negative column order.

The following theorem appeared in [9], which is necessary for our algorithm, states that

Theorem 1. [9] Let f be a monic bracket monomial, then the leading term of the normal form f under the negative column order is $f^{\downarrow\downarrow}$, where $f^{\downarrow\downarrow}$ is a standard bracket monomial by rewriting each column of f in an ascending way.

This result has been appeared in [5] in the double tableau form (Theorem 5). He considered Rota's original order: degree lexicographical column order, i.e., change (2) in definition 2 into the opposite way.

Example 2. $f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_4 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_6 \end{bmatrix}$, $f^{\downarrow\downarrow} = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}$, and from the normal form of f

$$f = \underbrace{\begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_3 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix},$$

we can also figure out that the leading term is $f^{\downarrow\downarrow}$.

As a corollary of theorem 1, we have

Corollary 1. [9] *The negative column order is an admissible order among straight bracket monomials.*

Next, we recall the theory of coordinate ring of bracket algebra. Assume that

$$\mathbf{a}_i = (x_{i1}, x_{i2}, \dots, x_{in})^T, 1 \leq i \leq m. \quad (2.3)$$

Then $\mathbb{K}[\{\mathbf{x}_{ij} : 1 \leq i \leq m, 1 \leq j \leq n\}]$ or just $\mathbb{K}[\{\mathbf{x}_{ij}\}]$ will be called the *coordinate ring* of the bracket algebra. So, we can define an expanding map from $\mathcal{B}[\{\mathbf{a}_i\}]$ to $\mathbb{K}[\{\mathbf{x}_{ij}\}]$. For any bracket polynomial f , the expanding form of f is denoted as f^c .

The corresponding order called *negative basis order* of the negative column order in the ring $\mathbb{K}[\{\mathbf{x}_{ij}\}]$ is as follows:

$$\begin{array}{ccccccc} x_{1n} & \succ & x_{2n} & \succ & \dots & \succ & x_{mn} \\ \succ & x_{1(n-1)} & \succ & x_{2(n-1)} & \succ & \dots & \succ & x_{m(n-1)} \\ \vdots & & \vdots & & \vdots & & \vdots & \\ \succ & x_{11} & \succ & x_{21} & \succ & \dots & \succ & x_{m1} \end{array} \quad (2.4)$$

Corollary 2. *Let f, g be two straight bracket monomials, denote $LM(f^c), LM(g^c)$ as the leading monomial of f^c, g^c in $\mathbb{K}[\{\mathbf{x}_{ij}\}]$ under the degree lexicographical order generated by the negative basis order (2.4). Then*

$$f = g \iff LM(f^c) = LM(g^c). \quad (2.5)$$

Corollary 3. [9] *For any monomial M of $\mathbb{K}[\{\mathbf{x}_{ij}\}]$ in the form*

$$\begin{array}{c} x_{i_1^1,1} x_{i_1^2,1} \dots x_{i_1^d,1} \\ x_{i_2^1,2} x_{i_2^2,2} \dots x_{i_2^d,2} \\ \vdots \\ x_{i_n^1,n} x_{i_n^2,n} \dots x_{i_n^d,n}, \end{array} \quad (2.6)$$

where $1 \leq i_1^j < i_2^j < \dots < i_n^j \leq n (j = 1, 2, \dots, d)$. Then there is a unique

bracket monomial f such that $LM(f^c) = M$. Indeed, $f = \begin{bmatrix} \mathbf{a}_{i_1^1}^1 & \mathbf{a}_{i_1^2}^1 & \dots & \mathbf{a}_{i_1^d}^1 \\ \mathbf{a}_{i_1^1}^2 & \mathbf{a}_{i_1^2}^2 & \dots & \mathbf{a}_{i_1^d}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_{i_n^1}^d & \mathbf{a}_{i_n^2}^d & \dots & \mathbf{a}_{i_n^d}^d \end{bmatrix}$.

Algorithm 1 Straightening algorithm based on coordinate**Input:** a non standard bracket monomial f **Output:** the normal form of f

- 1: compute f^c , and find the leading term $\lambda_1 M_1$ of f^c , where λ_1 is the coefficient. Construct a bracket monomial f_1 such that $LM(f_1^c) = M_1$.
- 2: compute f_1^c , and find the leading term $\lambda_2 M_2$ of $f^c - \lambda_1 f_1^c$. Construct a bracket monomial f_2 such that $LM(f_2^c) = M_2$.
- 3: compute f_2^c , and find the leading term $\lambda_3 M_3$ of $f^c - \lambda_1 f_1^c - \lambda_2 f_2^c$. Construct a bracket monomial f_3 such that $LM(f_3^c) = M_3$.
- 4: continue the above process, and finally we find $\lambda_1 f_1, \lambda_2 f_2, \dots, \lambda_s f_s$.
- 5: **return** $\lambda_1 f_1 + \lambda_2 f_2 + \dots + \lambda_s f_s$.

Remark 1. Since we know that the normal form $\sum \mu_i g_i$ (in a descending form under the negative column order) of a bracket monomial f always exists.

1. $\mu_1 = \lambda_1 = 1, f_1 = g_1 = f^{\downarrow\downarrow}$;
2. Assume that the leading term of $f^c - \lambda_1 f_1^c$ is $\lambda_2 M_2$. Then if $M_2 \succ LM(g_2^c)$, then we can never find M_2 in $\sum \mu_i g_i^c$, a contradiction. If $M_2 \prec LM(g_2^c)$, then we will never find $LM(g_2^c)$ in the expression $f^c - \lambda_1 f_1^c$. Thus $M_2 = LM(g_2^c)$, i.e., $f_2 = g_2$.

Therefore, this algorithm works.

3 Introduction of several straightening algorithms

The following are a basic introduction of the first three straightening algorithms introduced in section 1. We will not focus on the last one.

1. van der Waerden straightening algorithm [8], [14], [16]:

The van der Waerden straightening algorithm is based on the following syzygy, called *van der Waerden syzygy* (see [17] for more vivid description):

$$\sum_{\substack{\sigma \in S_{n+1} \\ \sigma(1) < \dots < \sigma(r) \\ \sigma(r+1) < \dots < \sigma(n+1)}} \text{sign}(\sigma) \begin{bmatrix} \mathbf{b}_1 & \dots & \mathbf{b}_{r-1} & \mathbf{c}_{\sigma(r+1)} & \mathbf{c}_{\sigma(r+2)} & \dots & \mathbf{c}_{\sigma(n+1)} \\ \mathbf{c}_{\sigma(1)} & \dots & \mathbf{c}_{\sigma(r-1)} & \mathbf{c}_{\sigma(r)} & \mathbf{d}_1 & \dots & \mathbf{d}_{n-r} \end{bmatrix} = 0. \quad (3.1)$$

Example 3. $f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_5 \end{bmatrix}.$

$$\begin{aligned} f &= \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_6 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_3 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} \end{aligned}$$

In the above example, in the second column, we have $\mathbf{a}_4 \succ \mathbf{a}_3$, so this is called a *violation*. Thus we can take $\mathbf{b}_1 = \mathbf{a}_1; \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4 = \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4, \mathbf{a}_6; \mathbf{d}_1 = \mathbf{a}_5$, and do (3.1) on f .

As for bracket monomials of degree large than two, we can take the *first-first strategy* [17] to do straightening: find the first pair of adjacent rows, starting from the top, which has a violation. Then choose the first column, starting from the left, that contains a violation in these two rows.

2. N. White's implementation of the straightening algorithm [17]:

The following is another syzygy called *multiple syzygy* used by N. White. They also play the role of Gröbner basis in the bracket algebra, just like van der Waerden syzygies.

$$\sum_{i_1 < \dots < i_{n-r}} \pm \begin{bmatrix} \mathbf{b}_1 \cdots \mathbf{b}_r & \mathbf{c}_{i_1} \cdots \mathbf{c}_{i_{n-r}} \\ \mathbf{c}_1 \cdots \mathbf{c}_n \leftarrow \mathbf{b}_{r+1} \cdots \mathbf{b}_n \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \cdots \mathbf{b}_n \\ \mathbf{c}_1 \cdots \mathbf{c}_n \end{bmatrix}, \quad (3.2)$$

where the notation “ \leftarrow ” refers to the substitution of $\mathbf{b}_{r+1}, \dots, \mathbf{b}_n$ for $\mathbf{c}_{i_1}, \dots, \mathbf{c}_{i_{n-r}}$ in $\mathbf{c}_1, \dots, \mathbf{c}_n$ in order. And the sign ± 1 depends on the permutation sign of the sequence $\mathbf{c}_1, \dots, \mathbf{c}_n \leftarrow \mathbf{b}_{r+1}, \dots, \mathbf{b}_n, \mathbf{c}_{i_1}, \dots, \mathbf{c}_{i_{n-r}}$ with respect to the sequence $\mathbf{c}_1, \dots, \mathbf{c}_n, \mathbf{b}_{r+1}, \dots, \mathbf{b}_n$.

Example 4. f is in example 3. Choose $r = 1, \mathbf{c}_1 = \mathbf{a}_2, \mathbf{c}_2 = \mathbf{a}_3, \mathbf{c}_3 = \mathbf{a}_5, \mathbf{b}_2 = \mathbf{a}_4, \mathbf{b}_3 = \mathbf{a}_6$. Then by (3.2), in one step, we get

$$f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}.$$

We find the first pair of adjacent two rows having a violation. And we only consider this two rows in the following implementation.

1. (a) If there is a violation in the last column, while there is no violation in the second column, then use the *multiply syzygy* on the last element of the former row and the entire latter row.
- (b) If there exist violations in the second and last columns, then use the *multiple syzygy* on the first element of the former row and the entire latter row.
- (c) If there is no violation in the last column, then use *van der Waerden syzygy* to correct the first violation.
2. Assume that the s -th column is the first column having a violation.
 - (a) If the last column or two of the last three columns have a violation, then use the *multiple syzygy* on the $(n - s + 1)$ last element of the former row and the entire latter row.
 - (b) Otherwise, use *van der Waerden syzygy* to correct this violation.
3. This is the same as 2, except that if there is a violation in the last column. The elements in the second row is denoted as \mathbf{b} . Finding the least one \mathbf{c} (say in the t -th row) in the last column situated below \mathbf{b} . Then use the *multiple syzygy* on the element \mathbf{c} of t -th row and the entire row of \mathbf{b} 's.

4. This is the same as 3, except that only the last column non-consecutive row multiple syzygies and the van der Waerden syzygies are used, not the multiple syzygies when there are violations in two of the last three columns.
5. Find the last violation, and use *van der Waerden syzygy* syzygies.

In example 4, we use the method of (2.a). Similarly, if we follow the idea of (1.b), then also in one step, we get the result.

3. Rota's straightening algorithm [4]:

The origin idea about this can be found in [4]. But here we should make a little changes such that it is more suitable. Let f be a Young tableau. Let α_i be the number of \mathbf{a}_i appeared in f . And we call

$$(\alpha) = (\alpha_1, \alpha_2, \dots, \alpha_n) \quad (3.3)$$

the *content* of f . Let $\{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n\}$ be some other n vectors in \mathcal{V}^n , for any $1 \leq l \in \mathbb{N}$ and $\mathbf{b}_j, \mathbf{a}_i$, the *set polarization operator*

$$\mathcal{D}^l(\mathbf{b}_j, \mathbf{a}_i) : \mathcal{B}[\{\mathbf{a}_i, \mathbf{b}_j\}] \longrightarrow \mathcal{B}[\{\mathbf{a}_i, \mathbf{b}_j\}] \quad (3.4)$$

with respect to $l, \mathbf{b}_j, \mathbf{a}_i$ is defined as:

1. if $\alpha_i < l$, then $\mathcal{D}^l(\mathbf{b}_j, \mathbf{a}_i)(f) := 0$;
2. if $\alpha_i \geq l$, then $\mathcal{D}^l(\mathbf{b}_j, \mathbf{a}_i)(f) := \sum_r f_r$, where f_r are all the $\binom{\alpha_i}{l}$ distinct monomials obtained from f by replacing each subset of l letters \mathbf{a}_i by l letters \mathbf{b}_j .

Example 5. $f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_6 \mathbf{a}_8 \end{bmatrix}$, then

$$\mathcal{D}^2(\mathbf{b}_1, \mathbf{a}_2) = \begin{bmatrix} \mathbf{a}_1 \mathbf{b}_1 \mathbf{a}_4 \\ \mathbf{b}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_6 \mathbf{a}_8 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{b}_1 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{b}_1 \mathbf{a}_6 \mathbf{a}_8 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{b}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{b}_1 \mathbf{a}_6 \mathbf{a}_8 \end{bmatrix}.$$

The set polarization operators commute with each other, and they are the building blocks of the *Capelli operator*, which is defined as

$$\mathcal{C}_f = \prod_{1 \leq q \leq n} \prod_{1 \leq i \leq m} \mathcal{D}^{\eta_i(q)}(\mathbf{b}_q, \mathbf{a}_i), \quad (3.5)$$

where $\eta_i(q)$ is the number of occurrences of \mathbf{a}_i in the q -th column of f .

Example 6. $f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_5 \mathbf{a}_6 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_7 \\ \mathbf{a}_3 \mathbf{a}_6 \mathbf{a}_8 \end{bmatrix}$, then

$$\begin{aligned} \mathcal{C}_f = & \mathcal{D}^1(\mathbf{b}_1, \mathbf{a}_1) \mathcal{D}^1(\mathbf{b}_1, \mathbf{a}_2) \mathcal{D}^1(\mathbf{b}_1, \mathbf{a}_3) \mathcal{D}^1(\mathbf{b}_2, \mathbf{a}_3) \mathcal{D}^1(\mathbf{b}_2, \mathbf{a}_5) \\ & \mathcal{D}^1(\mathbf{b}_2, \mathbf{a}_6) \mathcal{D}^1(\mathbf{b}_3, \mathbf{a}_6) \mathcal{D}^1(\mathbf{b}_3, \mathbf{a}_8) \mathcal{D}^1(\mathbf{b}_3, \mathbf{a}_9). \end{aligned}$$

So $\mathcal{C}_f(f) = \begin{bmatrix} \mathbf{b}_1 \mathbf{b}_2 \mathbf{b}_3 \\ \mathbf{b}_1 \mathbf{b}_2 \mathbf{b}_3 \\ \mathbf{b}_1 \mathbf{b}_2 \mathbf{b}_3 \end{bmatrix} = \mathcal{C}_f(f^{\downarrow\downarrow})$. If $f_1 = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_3 \mathbf{a}_6 \mathbf{a}_7 \\ \mathbf{a}_5 \mathbf{a}_6 \mathbf{a}_8 \end{bmatrix}$, then $\mathcal{C}_f(f_1) = 0$.

The main theorem about Capelli operator says that,

Theorem 2. [4] *Let f and g be two straight bracket monomials with the same content, then*

- (1). $\mathcal{C}_f(f) \neq 0$;
- (2). *If $f \succ g$ in the negative column order, then $\mathcal{C}_f(g) = 0$.*

From this theorem, we have

Corollary 4. [4] *Assume that $\{f_i\}$ is the set of all straight bracket monomials that have the same content of f . Set*

$$f = \sum \lambda_i f_i \quad (3.6)$$

where $f_1 \succ f_2 \succ \dots$ in the negative column order. Then operating \mathcal{C}_{f_i} on (3.6), we will get a down-triangular linear equations about $\lambda_1, \lambda_2, \dots$.

Example 7. f is in example 3. All the straight bracket monomials with the same content of f are

$$\begin{aligned} f_1 &= \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}, f_2 = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}, f_3 = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}, \\ f_4 &= \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_3 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}, f_5 = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} \end{aligned}$$

An easy computation shows that the down-triangle linear equation is

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ -1 \\ 0 \\ 0 \end{pmatrix}.$$

Hence, $\lambda_1 = 1, \lambda_2 = 0, \lambda_3 = -1, \lambda_4 = 0, \lambda_5 = -1$. Therefore,

$$f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}.$$

4 Column brackets and the new straightening algorithm

In this section, we will introduce the concept of column bracket, which is a new concept that comes from the relationship between the bracket algebra and it's coordinate ring. As an application, we will present a new straightening algorithm. We first recall two definitions from linear algebra [11].

Definition 3. (1). Let $A = (u_{ij})_{n \times n}$ be a matrix, then the permanent of A is defined as

$$\text{perm}(A) = \sum_{\sigma \in S_n} u_{1\sigma(1)} u_{2\sigma(2)} \cdots u_{n\sigma(n)}.$$

(2). Let $B = (v_{ij})_{r \times s}$, $C = (w_{ij})_{r \times s}$ be two matrices, then the Hadamard product of B, C is defined as

$$B \circ C = (v_{ij} w_{ij})_{r \times s}.$$

In the following, we list some properties of permanent and Hadamard product.

Proposition 1. (1). $\text{perm}(A)$ is invariant if we change two columns entirely;
(2). Hadamard product is a commutative operation.

The following is the definition of column bracket.

Definition 4. Let $T = \begin{bmatrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \cdots & \mathbf{b}_{1n} \\ \mathbf{b}_{21} & \mathbf{b}_{22} & \cdots & \mathbf{b}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{b}_{d1} & \mathbf{b}_{d2} & \cdots & \mathbf{b}_{dn} \end{bmatrix}$ be a bracket monomial of degree d ,

where $\{\mathbf{b}_{ij} : 1 \leq i \leq d, 1 \leq j \leq n\} \subseteq \{\mathbf{a}_1, \dots, \mathbf{a}_m\}$. Then we define

$$\begin{aligned} \langle T \rangle &:= \text{perm}(\mathbf{b}_{11} \circ \mathbf{b}_{21} \cdots \circ \mathbf{b}_{d1}, \mathbf{b}_{12} \circ \mathbf{b}_{22} \cdots \circ \mathbf{b}_{d2}, \dots, \mathbf{b}_{1n} \circ \mathbf{b}_{2n} \cdots \circ \mathbf{b}_{dn}), \\ \{T\} &:= [\mathbf{b}_{11} \circ \mathbf{b}_{21} \cdots \circ \mathbf{b}_{d1}, \mathbf{b}_{12} \circ \mathbf{b}_{22} \cdots \circ \mathbf{b}_{d2}, \dots, \mathbf{b}_{1n} \circ \mathbf{b}_{2n} \cdots \circ \mathbf{b}_{dn}]. \end{aligned} \quad (4.1)$$

The first one is called even column bracket, while the second one is called odd column bracket. We call d the degree of the column bracket.

Example 8. (1). Let $A = \begin{pmatrix} x_1 & x_2 \\ x_3 & x_4 \end{pmatrix}$, then $\text{perm}(A) = x_1 x_4 + x_2 x_3$.

(2). Let $B = \begin{pmatrix} y_1 & y_2 \\ y_3 & y_4 \end{pmatrix}$, then $A \circ B = \begin{pmatrix} x_1 y_1 & x_2 y_2 \\ x_3 y_3 & x_4 y_4 \end{pmatrix}$.

(3). $n = d = 2$. Then

$$\begin{Bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \\ \mathbf{a}_3 & \mathbf{a}_4 \end{Bmatrix} = x_1 x_3 y_2 y_4 + x_2 x_4 y_1 y_3, \quad \left\langle \begin{Bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \\ \mathbf{a}_3 & \mathbf{a}_4 \end{Bmatrix} \right\rangle = x_1 x_3 y_2 y_4 - x_2 x_4 y_1 y_3.$$

Proposition 2. (1). An even column bracket contains $n!$ terms, which is invariant if we permute two elements in a column, and also invariant under the permutation of any two columns entirely.

(2). An odd column bracket contains $n!$ terms, which is invariant if we permute two elements in a column, while anti-invariant under the permutation of any two columns entirely.

Proof. Just by the definitions of column brackets and the proposition 1. \square

Due to (1) and (2) of this proposition, we can always make some permutations such that the column bracket satisfies each column is non-descending and the first row is non-descending. And in the following, when refers to column brackets, we always suppose that it is in this form. The following proposition depicts the multiplication rules among column brackets.

Proposition 3. (*Multiplication Rules*) Let

$$A = (\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_n), B = (\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_n)$$

be two tableaux of size $k \times n$ and $l \times n$ of \mathbf{a} 's, respectively. Then the following identities hold:

$$\begin{aligned} (1). \langle A \rangle \cdot \langle B \rangle &= \sum_{\sigma \in S_n} \left\langle \begin{matrix} A \\ \sigma(B) \end{matrix} \right\rangle = \sum_{\sigma \in S_n} \left\langle \begin{matrix} \sigma(A) \\ B \end{matrix} \right\rangle, \\ (2). \langle A \rangle \cdot \{B\} &= \sum_{\sigma \in S_n} \text{sign}(\sigma) \left\{ \begin{matrix} A \\ \sigma(B) \end{matrix} \right\} = \sum_{\sigma \in S_n} \left\{ \begin{matrix} \sigma(A) \\ B \end{matrix} \right\}, \\ (3). \{A\} \cdot \{B\} &= \sum_{\sigma \in S_n} \text{sign}(\sigma) \left\langle \begin{matrix} A \\ \sigma(B) \end{matrix} \right\rangle = \sum_{\sigma \in S_n} \text{sign}(\sigma) \left\langle \begin{matrix} \sigma(A) \\ B \end{matrix} \right\rangle, \end{aligned}$$

where

$$\sigma(A) = (\mathbf{A}_{\sigma(1)}, \mathbf{A}_{\sigma(2)}, \dots, \mathbf{A}_{\sigma(n)}), \sigma(B) = (\mathbf{B}_{\sigma(1)}, \mathbf{B}_{\sigma(2)}, \dots, \mathbf{B}_{\sigma(n)}).$$

Proof. We only give a proof the second identity of (1). The others can be proved similarly. For convenience, only here, we set $A = (\mathbf{a}_{ij})_{k \times n}$, $B = (\mathbf{b}_{ij})_{l \times n}$, and $\mathbf{a}_{ij} = (x_{ij,1}, x_{ij,2}, \dots, x_{ij,n})$, $\mathbf{b}_{ij} = (y_{ij,1}, y_{ij,2}, \dots, y_{ij,n})$. Then

$$\begin{aligned} &L.H.S. \\ &= \left(\sum_{\sigma \in S_n} x_{11,\sigma(1)} x_{21,\sigma(1)} \cdots x_{k1,\sigma(1)} \cdots x_{1n,\sigma(n)} x_{2n,\sigma(n)} \cdots x_{kn,\sigma(n)} \right) \\ &\quad \times \left(\sum_{\tau \in S_n} y_{11,\tau(1)} y_{21,\tau(1)} \cdots y_{k1,\tau(1)} \cdots y_{1n,\tau(n)} y_{2n,\tau(n)} \cdots y_{kn,\tau(n)} \right) \\ &= \sum_{\rho \in S_n} \left\{ \sum_{\tau \in S_n} \left(x_{1\rho(1),\tau(1)} x_{2\rho(1),\tau(1)} \cdots x_{k\rho(1),\tau(1)} \cdots x_{1\rho(n),\tau(n)} x_{2\rho(n),\tau(n)} \cdots x_{k\rho(n),\tau(n)} \right) \right. \\ &\quad \left. \times \left(y_{11,\tau(1)} y_{21,\tau(1)} \cdots y_{k1,\tau(1)} \cdots y_{1n,\tau(n)} y_{2n,\tau(n)} \cdots y_{kn,\tau(n)} \right) \right\} \\ &= R.H.S. \end{aligned}$$

□

This proposition is a reason why we call $\langle T \rangle$ even and $\{T\}$ odd. The following proposition is our original intention about the introduction of column bracket.

Theorem 3. Let $T = \begin{bmatrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \cdots & \mathbf{b}_{1n} \\ \mathbf{b}_{21} & \mathbf{b}_{22} & \cdots & \mathbf{b}_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{b}_{d1} & \mathbf{b}_{d2} & \cdots & \mathbf{b}_{dn} \end{bmatrix}$ be a bracket monomial with degree d , then

$$\begin{aligned} T^c &= \sum_{\sigma_2, \dots, \sigma_d \in S_n} \text{sign}(\sigma) \left\langle \begin{matrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \cdots & \mathbf{b}_{1n} \\ \mathbf{b}_{2\sigma_2(1)} & \mathbf{b}_{2\sigma_2(2)} & \cdots & \mathbf{b}_{2\sigma_2(n)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{b}_{d\sigma_d(1)} & \mathbf{b}_{d\sigma_d(2)} & \cdots & \mathbf{b}_{d\sigma_d(n)} \end{matrix} \right\rangle, \text{ if } d \text{ is even;} \\ T^c &= \sum_{\sigma_2, \dots, \sigma_d \in S_n} \text{sign}(\sigma) \left\{ \begin{matrix} \mathbf{b}_{11} & \mathbf{b}_{12} & \cdots & \mathbf{b}_{1n} \\ \mathbf{b}_{2\sigma_2(1)} & \mathbf{b}_{2\sigma_2(2)} & \cdots & \mathbf{b}_{2\sigma_2(n)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{b}_{d\sigma_d(1)} & \mathbf{b}_{d\sigma_d(2)} & \cdots & \mathbf{b}_{d\sigma_d(n)} \end{matrix} \right\}, \text{ if } d \text{ is odd.} \end{aligned} \quad (4.2)$$

where $\text{sign}(\sigma) = \text{sign}(\sigma_2) \cdots \text{sign}(\sigma_k)$. The right hand side of (4.2) will be denoted as $\mathcal{P}_c(T)$.

Proof. This is just a corollary of proposition 3. \square

This proposition is another reason why we call $\langle T \rangle$ even and $\{T\}$ odd.

Definition 5. Let T be a bracket monomial. Assume that $\mathcal{P}_c(T) = \sum \lambda_i \{T_i\}$ or $= \sum \lambda_i \langle T_i \rangle$. Define a set \mathcal{S} about the terms of $\mathcal{P}_c(T)$ as

$$\mathcal{S} := \{T_i : \text{as a bracket monomial, } [T_i] \text{ is straight}\}. \quad (4.3)$$

Then define

$$\mathcal{C}(T) := \sum_{T_i \in \mathcal{S}} \lambda_i [T_i] \quad (4.4)$$

as a straight bracket polynomial. Furthermore, if $f = \sum \lambda_i T_i$ be a bracket polynomial, then define

$$\mathcal{C}(f) = \sum \lambda_i \mathcal{C}(T_i). \quad (4.5)$$

Example 9. Let $T = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}$, then

$$\begin{aligned} \mathcal{P}_c(T) &= \left\langle \begin{matrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{matrix} \right\rangle - \left\langle \begin{matrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_6 \mathbf{a}_5 \end{matrix} \right\rangle - \left\langle \begin{matrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_4 \mathbf{a}_3 \mathbf{a}_6 \end{matrix} \right\rangle + \left\langle \begin{matrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{matrix} \right\rangle \\ &\quad + \left\langle \begin{matrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_6 \mathbf{a}_3 \mathbf{a}_5 \end{matrix} \right\rangle - \left\langle \begin{matrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_6 \mathbf{a}_5 \mathbf{a}_4 \end{matrix} \right\rangle. \\ \mathcal{C}(T) &= \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}. \end{aligned}$$

Remark 2. If T is straight, then it is the leading term of $\mathcal{C}(T)$ under the negative column order. In the following, we always write $\mathcal{C}(T)$ in an descending form under the negative column order.

Proposition 3 is the basis of our straightening algorithm. Let's first see an example as an introduction.

Example 10. Let $f = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_5 \end{bmatrix}$, we want to compute the straight formula of f . Based on the thought of coordinate, we can do it like as follows.

Step 1. By theorem 1, we know that the leading term of f is

$$f_0 = f^{\downarrow\downarrow} = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix},$$

so we can set $f = f_0 + g_1$, and it suffices to get the straight expression of g_1 .

Step 2. We only need to find the leading term of g_1 , denoted f_1 . Since

$$\begin{aligned} \mathcal{P}_c(f) - \mathcal{P}_c(f_0) = & - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} \right\rangle + \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_4 \mathbf{a}_3 \mathbf{a}_6 \end{bmatrix} \right\rangle + \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_3 \mathbf{a}_6 \mathbf{a}_5 \end{bmatrix} \right\rangle - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_6 \mathbf{a}_3 \mathbf{a}_5 \end{bmatrix} \right\rangle \\ & + \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_5 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} \right\rangle - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_5 \mathbf{a}_6 \mathbf{a}_4 \end{bmatrix} \right\rangle - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} \right\rangle + \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_6 \mathbf{a}_5 \mathbf{a}_4 \end{bmatrix} \right\rangle. \end{aligned}$$

So,

$$f_1 = - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix},$$

and we get $f = f_0 + f_1 + g_2$.

Step 3. Find the leading term f_2 of g_2 . Since

$$\begin{aligned} \mathcal{P}_c(f) - \mathcal{P}_c(f_0) - \mathcal{P}_c(f_1) = & \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_5 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} \right\rangle - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_5 \mathbf{a}_6 \mathbf{a}_4 \end{bmatrix} \right\rangle - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} \right\rangle \\ & + \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_6 \mathbf{a}_5 \mathbf{a}_4 \end{bmatrix} \right\rangle + \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_6 \mathbf{a}_5 \end{bmatrix} \right\rangle - \left\langle \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_6 \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix} \right\rangle. \end{aligned}$$

So

$$f_2 = - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix},$$

and hence we get $f = f_0 + f_1 + f_2$. Note that $\mathcal{P}_c(f) - \mathcal{P}_c(f_0) - \mathcal{P}_c(f_1) - \mathcal{P}_c(f_2) = 0$, therefore,

$$\begin{bmatrix} \mathbf{a}_1 \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_5 \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}.$$

We can generalize this process into the following algorithm 2. This algorithm is a little similar to Algorithm 1, while a little better than Algorithm 1. Because if the bracket monomial T has degree d , then T^c has $(n!)^d$ terms, while $\mathcal{P}_c(T)$ has $(n!)^{d-1}$ terms. As we can see, this is also not a good straightening algorithm.

Algorithm 2 Straightening algorithm based on column bracket expanding**Input:** a non standard bracket monomial f **Output:** the normal form of f

- 1: compute $\mathcal{P}_c(f) - \mathcal{P}_c(f_0)$, where $f_0 = f^{\downarrow\downarrow}$;
- 2: find the largest term $\lambda_1\{f_1\}$ or $\lambda_1\langle f_1\rangle$ in $\mathcal{P}_c(f) - \mathcal{P}_c(f_0)$ under the negative column order, and compute $\mathcal{P}_c(f) - \mathcal{P}_c(f_0) - \lambda_1\mathcal{P}_c(f_1)$;
- 3: find the largest term $\lambda_2\{f_2\}$ or $\lambda_2\langle f_2\rangle$ in $\mathcal{P}_c(f) - \mathcal{P}_c(f_0) - \lambda_1\mathcal{P}_c(f_1)$, and compute $\mathcal{P}_c(f) - \mathcal{P}_c(f_0) - \lambda_1\mathcal{P}_c(f_1) - \lambda_2\mathcal{P}_c(f_2)$;
- 4: continue the above process, and finally we will find $f_0, \lambda_1 f_1, \lambda_2 f_2, \dots, \lambda_s f_s$;
- 5: **return** $f_0 + \lambda_1 f_1 + \lambda_2 f_2 + \dots + \lambda_s f_s$.

Remark 3. (1). We can similarly to define the negative column order among column brackets, just as that in bracket algebra. At this time, it is just a total order, but not an admissible order anymore.

(2). As we know, a determinant or a permanent is uniquely determined by any monomial of its expression. Thus, the column bracket decomposition (4.2) is equivalent to the coordinate expression. Hence, algorithm 2 works.

In Algorithm 2, via computing $\mathcal{P}_c(f), \mathcal{P}_c(f_i)$ we find the straight expression of f . This is almost the same way of Algorithm 1. So, when the degree of f and the dimension of bracket algebra increase, this method is impractical. Indeed, we do not need to compute $\mathcal{P}_c(f_i)$, instead it suffices to compute $\mathcal{C}(f_i)$. So we have the following new algorithm, whose validity is based on invariant division and column bracket.

Algorithm 3 Straightening algorithm based on column bracket**Input:** a non standard bracket monomial f **Output:** the normal form of f

- 1: compute $\mathcal{C}(f) - \mathcal{C}(f_0)$, where $f_0 = f^{\downarrow\downarrow}$;
- 2: find the largest term $\lambda_1 f_1$ in $\mathcal{C}(f) - \mathcal{C}(f_0)$ under the negative column order, and compute $\mathcal{C}(f) - \mathcal{C}(f_0) - \lambda_1 \mathcal{C}(f_1)$;
- 3: find the largest term $\lambda_2 f_2$ in $\mathcal{C}(f) - \mathcal{C}(f_0) - \lambda_1 \mathcal{C}(f_1)$, and compute $\mathcal{C}(f) - \mathcal{C}(f_0) - \lambda_1 \mathcal{C}(f_1) - \lambda_2 \mathcal{C}(f_2)$;
- 4: continue the above process, and finally we will find $f_0, \lambda_1 f_1, \lambda_2 f_2, \dots, \lambda_s f_s$;
- 5: **return** $f_0 + \lambda_1 f_1 + \lambda_2 f_2 + \dots + \lambda_s f_s$.

Remark 4. (1). Let T be a bracket monomial with degree d , and let N be the number of all straight Young tableaux with a lower negative column order than $f^{\downarrow\downarrow}$, then the costs of algorithm 3 is at most

$$N(n!)^{d-1} = e^{\log N + (d-1) \sum_{i=1}^n \log i}. \quad (4.6)$$

(2). Let $f = \sum \lambda_i T_i$ be a bracket polynomial, then straight f can be achieved by first computing $\mathcal{C}(f) = \sum \lambda_i \mathcal{C}(T_i)$, then finding the leading term $\mu_1 S_1$ of $\mathcal{C}(f)$ and compute $\mathcal{C}(f) - \mu_1 \mathcal{C}(S_1)$, and continue this procedure like algorithm 3.

So, Algorithm 3 reduces to the computation of $\mathcal{C}(T)$. As for the computation of $\mathcal{C}(T)$, we first present an example to make our idea clear.

$$\begin{aligned}
& \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 \mathbf{a}_4 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_6 \end{bmatrix}\right) \\
&= \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_2 & \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \mathbf{a}_6 \\ \mathbf{a}_3 & \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix}\right) \\
&= \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 & \mathbf{a}_5 \\ \mathbf{a}_2 & \mathbf{a}_4 & \mathbf{a}_6 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 & \mathbf{a}_4 \\ \mathbf{a}_2 & \mathbf{a}_5 & \mathbf{a}_6 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_5 \\ \mathbf{a}_3 & \mathbf{a}_4 & \mathbf{a}_6 \end{bmatrix}\right) + \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_4 \\ \mathbf{a}_3 & \mathbf{a}_5 & \mathbf{a}_6 \end{bmatrix}\right) \\
&= \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_4 \\ \mathbf{a}_3 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}
\end{aligned}$$

where for example $\mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_2 & \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix}\right)$ is a bracket polynomial with the first column $\begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \end{bmatrix}$ and the remainder two columns are composed of the elements of $\mathcal{C}\left(\begin{bmatrix} \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix}\right)$. Similarly, we have $\mathcal{C}\left(\begin{bmatrix} \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix}\right) = \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_3 & \mathbf{a}_5 \\ \mathbf{a}_4 & \mathbf{a}_6 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_3 & \mathbf{a}_4 \\ \mathbf{a}_5 & \mathbf{a}_6 \end{bmatrix}\right) = \begin{bmatrix} \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}$ which is relatively easy to compute. Thus $\mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_2 & \mathbf{a}_4 \mathbf{a}_5 \end{bmatrix}\right) = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}$. This method can be easily extended to the general case, but we should be careful of the ± 1 signs, which is based on proposition 3.

We now make a remark on the first identity, the other identities are similarly to analyze. Because of the formulas (4.2), in order to compute $\mathcal{C}(T)$, we first should choose an element \mathbf{x} from the first row, and \mathbf{a}_1 is the smallest one, so $\mathbf{x} = \mathbf{a}_1$. In the second row, we want pick another element \mathbf{y} , since when choosing $\mathbf{a}_2, \mathbf{a}_3$, we definitely can make sure that the tableaux in the following construction are straight, so $\mathbf{y} = \mathbf{a}_2, \mathbf{a}_3$, but we can not choose $\mathbf{y} = \mathbf{a}_6$. Generalize this, we have the following algorithm of computing $\mathcal{C}(T)$.

Algorithm 4 A algorithm of computing $\mathcal{C}(T)$: multilinear case

Input: a bracket monomial T of degree d

Output: $\mathcal{C}(T)$

- 1: set \mathbf{x}_1 as the first element of the first row;
 - 2: find all possible $\mathbf{x}_2, \dots, \mathbf{x}_d$ by the following two rules **R1** and **R2**. Any such choice of $\mathbf{x}_2, \dots, \mathbf{x}_n$ combining \mathbf{x}_1 will construct a vector $[\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_d]^T$. Denote all such possible vectors as S_{11}, \dots, S_{1c_1} , they will construct the first column of elements in $\mathcal{C}(T)$. The corresponding remaining part will be denoted as T_{11}, \dots, T_{1c_1} ;
 - 3: continue step 1 and 2 for T_{11}, \dots, T_{1c_1} , and we can find $\mathcal{C}(T_{11}), \dots, \mathcal{C}(T_{1c_1})$;
 - 4: **return**
$$\sum_{\substack{i=1, \dots, c_1 \\ S'_{1i} \text{ is a term of } \mathcal{C}(T_{1i})}} \pm [S_{1i}, S'_{1i}].$$
-

Remark 5. (1). The ± 1 sign in the return of algorithm 4 can be computed from (4.2). We will not take any attention on it.

(2). As we find that, Algorithm 2 is similar to Algorithm 1, while Algorithm 3 is an optimization of Algorithm 2. This means, we can also give another algorithm which is similar to Algorithm 3, which is an optimization of Algorithm 1. However, the good thing for column bracket is that they have the same form of tableaux. This enables us to handle the algorithm in the system of column bracket more directly.

Assume that $T \neq 0$ is a bracket monomial of degree d , in n dimensional bracket algebra. The vectors appeared in T denoted as $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_{dn}$, in order. The following are some rules how to choose $\mathbf{x}_2, \dots, \mathbf{x}_d$ in step 2 of algorithm 4.

R1 the largest $n-1$ vectors $\mathbf{b}_r, \dots, \mathbf{b}_{r-n+2}$ can not be chosen in the first column.

R2 if \mathbf{b}_i is chosen as the s -th vector in the first column, then $\alpha_1 + \dots + \alpha_{i-1} \leq (s-1)n$.

R3 \mathbf{b}_i can not lies in the same rows of $\mathbf{b}_1, \dots, \mathbf{b}_{i-1}$.

Example 11. $T = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_8 \mathbf{a}_9 \\ \mathbf{a}_2 \mathbf{a}_5 \mathbf{a}_7 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}$. Then by **R1**, $\mathbf{a}_8, \mathbf{a}_9$ can not be chosen in the first column, by **R2**, the second element of the first column can not be $\mathbf{a}_5, \mathbf{a}_6$. Thus, we have

$$\begin{aligned} \mathcal{C}(T) = & \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_2 & \mathbf{a}_5 \mathbf{a}_7 \\ \mathbf{a}_3 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_2 & \mathbf{a}_5 \mathbf{a}_7 \\ \mathbf{a}_4 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) + \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 \mathbf{a}_4 \\ \mathbf{a}_2 & \mathbf{a}_5 \mathbf{a}_7 \\ \mathbf{a}_6 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \mathbf{a}_7 \\ \mathbf{a}_3 & \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_5 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) \\ & + \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 & \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_7 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) + \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \mathbf{a}_7 \\ \mathbf{a}_4 & \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_5 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) - \mathcal{C}\left(\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_4 & \mathbf{a}_3 \mathbf{a}_6 \\ \mathbf{a}_7 & \mathbf{a}_8 \mathbf{a}_9 \end{bmatrix}\right) \end{aligned}$$

Remark 6. Indeed, **R1** is included in **R2**, but it is the most obvious one, so we list it as a rule. By the way, we can find that **R2** is a rule how can we list all the straight tableaux. Hence, each choice in step 2 of algorithm 4 will always generate an term of $\mathcal{C}(T)$, which means this algorithm is optimal for multilinear tableaux.

Example 12. f is in example 3.

Step 1, compute $\mathcal{C}(f)$ and $\mathcal{C}(f_0)$, where $f_0 = f^{\downarrow\downarrow}$,

$$\mathcal{C}(f) = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}, \quad \mathcal{C}(f_0) = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} + \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}.$$

Then

$$\mathcal{C}(f) - \mathcal{C}(f_0) = - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix},$$

thus $f_1 = - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} =$ the leading term of $\mathcal{C}(f) - \mathcal{C}(f_0)$ under the negative column order.

Step 2, compute $\mathcal{C}(f_1) = f_1$. So

$$\mathcal{C}(f) - \mathcal{C}(f_0) - \mathcal{C}(f_1) = - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix},$$

hence $f_2 = - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}.$

Step 3, compute $\mathcal{C}(f_2) = f_2$, and so $\mathcal{C}(f) - \mathcal{C}(f_0) - \mathcal{C}(f_1) - \mathcal{C}(f_2) = 0$. Therefore, we have

$$\begin{bmatrix} \mathbf{a}_1 \mathbf{a}_4 \mathbf{a}_6 \\ \mathbf{a}_2 \mathbf{a}_3 \mathbf{a}_5 \end{bmatrix} = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_5 \\ \mathbf{a}_3 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix} - \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}.$$

By comparing Rota's algorithm and our algorithm, we find that, the action of Capelli operator is just a substitution, and take the sum of the sign of all the possible substitutions as an element of the down-triangle matrix. While our algorithm do the permutations directly, and take care of the summation of the sign of all the suitable permutations. So we do not need to introduce additional n vectors, and find all the straight tableaux possessing the same content. A disadvantage of our algorithm is that if the straight expression of a bracket monomial contains k monomials, then the algorithm needs k steps.

Another drawback of our algorithm with respect to van der Waerden straightening algorithm or N. White's implementation algorithms, lies in the straightening of bracket polynomials f . We should first compute $\mathcal{C}(f)$, while van der Waerden straightening algorithm and N. White's implementation algorithms do not need this procedure.

In the following, we give an algorithm to compute $\mathcal{C}(T)$ in the general case.

Definition 6. Let $T = [\mathbf{b}_{ij}]_{m \times n}$ be a tableau, define

$$\mathcal{V}(T) := \{\mathbf{b}_1^{\alpha_1}, \mathbf{b}_2^{\alpha_2}, \dots, \mathbf{b}_k^{\alpha_k}\}, \quad (4.7)$$

as the set of all vectors that appeared in T , where α_i is the multiplicity of \mathbf{b}_i , and $\mathbf{b}_1 \prec \mathbf{b}_2 \prec \dots \prec \mathbf{b}_k$.

Definition 7. Let T be a tableau, \mathbf{b} is a vector appeared in T , define

$$\mathcal{R}(T, \mathbf{b}) := \{\mathcal{V}(\text{the row of } T \text{ that contains } \mathbf{b})\}. \quad (4.8)$$

Remark 7. The operation $\mathcal{V}(T_1) - \mathcal{V}(T_2)$ is seemed as the difference operation of sets.

Algorithm 5 A algorithm of computing $\mathcal{C}(T)$: general case**Input:** a bracket monomial T of degree m **Output:** $\mathcal{C}(T)$

- 1: Blocking T as $\begin{bmatrix} \mathbf{b}T_1 \\ T_2 \end{bmatrix}$, and let $S = \begin{bmatrix} \mathbf{b} S_1 \\ \mathbf{c} S_2 \end{bmatrix}$, where $\mathbf{b} = (\mathbf{b}_1, \dots, \mathbf{b}_1)^T, \mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_{\beta_1}), \alpha_1 + \beta_1 = m$.
- 2: $\mathbf{c}_1 \preceq \dots \preceq \mathbf{c}_{\beta_1} \in \mathcal{A} := \mathcal{V}(T_2) - \{\text{the last } (n-1) \text{ vectors of } \mathcal{V}(T_2)\} = \{\mathbf{a}_{11}^{\lambda_{11}}, \dots, \mathbf{a}_{1t_1}^{\lambda_{1t_1}}\}$
- 3: If $\mathbf{c}_1 = \mathbf{a}_{1i}$, then
 - 3.1: let $\{\mathbf{a}_{11}^{\mu_{11}}, \dots, \mathbf{a}_{1,i-1}^{\mu_{1,i-1}}\} = \{\mathbf{a}_{11}^{\lambda_{11}}, \dots, \mathbf{a}_{1,i-1}^{\lambda_{1,i-1}}\} \cup \mathcal{V}(T_1)$, so it must be

$$\mu_{11}, \dots, \mu_{1,i-1} \leq \alpha_1, \text{ and } \#\{\mathbf{a}_i : \mathbf{a}_i \prec \mathbf{c}_1\} \leq \alpha_1 n.$$

Otherwise, we can not choose such a \mathbf{c}_1 .

- 3.2: $\mathbf{c}_2, \dots, \mathbf{c}_{\beta_1} \in \mathcal{A}_1 := \{\mathbf{a}_i \in \mathcal{A} - \mathcal{V}(\mathcal{R}'(T, \mathbf{c}_1)) - \{\mathbf{a}_{11}^{\lambda_{11}}, \dots, \mathbf{a}_{1t_1}^{\lambda_{1,i-1}}\} : \mathbf{a}_i \succeq \mathbf{c}_1\}$, where $\mathcal{R}'(T, \mathbf{a}_{1,i}) \in \mathcal{R}(T, \mathbf{c}_1)$.
- 4: Assume that $\mathbf{c}_1, \dots, \mathbf{c}_s$ are all determined. Let $\mathcal{A}_s = \{\mathbf{a}_{s1}^{\lambda_{s1}}, \dots, \mathbf{a}_{st_s}^{\lambda_{st_s}}\}$.
- 5: If $\mathbf{c}_{s+1} = \mathbf{a}_{si}$, then
 - 5.1: Let $\{\mathbf{a}_{s1}^{\mu_{s1}}, \dots, \mathbf{a}_{s,i-1}^{\mu_{s,i-1}}\} = \{\mathbf{a}_{s1}^{\lambda_{s1}}, \dots, \mathbf{a}_{s,i-1}^{\lambda_{s,i-1}}\} \cup \mathcal{V}(T_1) \cup \mathcal{V}(T_2, \mathbf{a}_{si})$, where $\mathcal{V}(T_2, \mathbf{a}_{si})$ denotes to compute the collect of vectors appeared in the rows above the row of \mathbf{a}_{si} in T_2 . So it must be

$$\mu_{s1}, \dots, \mu_{s,i-1} \leq \alpha_1 + d, \text{ and } \#\{\mathbf{a}_i : \mathbf{a}_i \prec \mathbf{c}_{s+1}\} \leq (\alpha_1 + d)n.$$

where d is the number of rows that above the row of \mathbf{a}_{si} in T_2 . Otherwise, we can not choose such a \mathbf{c}_{s+1} .

- 5.2: $\mathbf{c}_{s+2}, \dots, \mathbf{c}_{\beta_1} \in \mathcal{A}_{s+1} := \{\mathbf{a}_i \in \mathcal{A}_s - \mathcal{V}(\mathcal{R}'(T, \mathbf{c}_{s+1})) - \{\mathbf{a}_{s1}^{\mu_{s1}}, \dots, \mathbf{a}_{s,i-1}^{\mu_{s,i-1}}\} : \mathbf{a}_i \succeq \mathbf{c}_{s+1}\}$, where $\mathcal{R}'(T, \mathbf{a}_{s,i}) \in \mathcal{R}(T, \mathbf{c}_{s+1})$ and different from the ones chosen above.
- 6: Finally, we will find all possible choices of $\mathbf{c}_1, \dots, \mathbf{c}_{\beta_1}$. Denote

$$\{\mathbf{b}_1, \dots, \mathbf{b}_1, \mathbf{c}_{11}, \dots, \mathbf{c}_{1\beta_1}\}, \dots, \{\mathbf{b}_1, \dots, \mathbf{b}_1, \mathbf{c}_{l1}, \dots, \mathbf{c}_{l\beta_1}\}$$

as column vectors $\mathbf{t}_1, \dots, \mathbf{t}_l$.

- 7: Using 1-5 to compute $\mathcal{C}(P_1), \dots, \mathcal{C}(P_l)$, where P_i is the tableau by deleting \mathbf{t}_i in T .
- 8: **return** $\sum_{\substack{i=1, \dots, l \\ P'_i \text{ is a term of } \mathcal{C}(P_i)}} \pm[\mathbf{t}_i, P'_i]$.

Remark 8. Among $\{\mathbf{c}_{i1}, \dots, \mathbf{c}_{i\beta_1}\}, i = 1, \dots, l$. If \mathbf{c}_{ij} lies in the w_{ij} -th column of T , and denote $s_i = w_{i1} + \dots + w_{i\beta_1} - \beta_1$. Then the ± 1 sign of $[\mathbf{t}_i, P'_i]$ is $(-1)^{s_i} \mu_i$, where μ_i is the coefficient of P'_i in $\mathcal{C}(P_i)$.

Example 13. $T = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \mathbf{a}_6 & \mathbf{a}_7 \\ \mathbf{a}_2 & \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_9 \\ \mathbf{a}_3 & \mathbf{a}_5 & \mathbf{a}_{10} & \mathbf{a}_{11} \\ \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_{10} & \mathbf{a}_{12} \\ \mathbf{a}_7 & \mathbf{a}_9 & \mathbf{a}_{11} & \mathbf{a}_{12} \end{bmatrix}$, and $S = \begin{bmatrix} \mathbf{a}_1 & \dots \\ \mathbf{c}_1 & \dots \\ \mathbf{c}_2 & \dots \\ \mathbf{c}_3 & \dots \\ \mathbf{c}_4 & \dots \end{bmatrix}$.

1. $\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4 \in \mathcal{A} = \{\mathbf{a}_2, \mathbf{a}_3^3, \mathbf{a}_5, \mathbf{a}_7, \mathbf{a}_8^2, \mathbf{a}_9^2\}$, $\beta_1 = 4$.
2. $\mathbf{c}_1 = \mathbf{a}_2$, then $\mathcal{R}(T, \mathbf{c}_1) = \{\mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_9\}$, so $\mathbf{c}_2, \mathbf{c}_3, \mathbf{c}_4 \in \mathcal{A}_1 = \{\mathbf{a}_3^2, \mathbf{a}_5, \mathbf{a}_7, \mathbf{a}_8, \mathbf{a}_9\}$.
3. $\mathbf{c}_2 = \mathbf{a}_3$ in the third row, then $\mathcal{R}(T, \mathbf{c}_2) = \{\mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_{10}, \mathbf{a}_{11}\}$, so $\mathbf{c}_3, \mathbf{c}_4 \in \mathcal{A}_2 = \{\mathbf{a}_3, \mathbf{a}_7, \mathbf{a}_8, \mathbf{a}_9\}$.
4. $\mathbf{c}_3 = \mathbf{a}_3$ in the fourth row, then $\mathcal{R}(T, \mathbf{c}_3) = \{\mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_{10}, \mathbf{a}_{12}\}$, so $\mathbf{c}_4 \in \mathcal{A}_3 = \{\mathbf{a}_7, \mathbf{a}_9\}$. Thus we get $[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_3, \mathbf{a}_7], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_3, \mathbf{a}_9]$.
5. if $\mathbf{c}_3 = \mathbf{a}_7$ is step 4, then $\mathcal{R}(T, \mathbf{c}_3) = \{\mathbf{a}_7, \mathbf{a}_9, \mathbf{a}_{11}, \mathbf{a}_{12}\}$, thus $\mathbf{c}_4 \in \mathcal{A}_3 = \{\mathbf{a}_8\}$. Therefore, we get $[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_7, \mathbf{a}_8]$.
6. if $\mathbf{c}_3 = \mathbf{a}_8$ is step 4, then $\mathcal{R}(T, \mathbf{c}_3) = \{\mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_{10}, \mathbf{a}_{12}\}$, thus $\mathbf{c}_4 \in \mathcal{A}_3 = \{\mathbf{a}_9\}$. Therefore, we get $[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_9]$.
7. if $\mathbf{c}_2 = \mathbf{a}_3$ in the fourth row in step 3, then $\mathcal{R}(T, \mathbf{c}_2) = \{\mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_{10}, \mathbf{a}_{12}\}$, therefore $\mathbf{c}_3, \mathbf{c}_4 \in \mathcal{A}_2 = \{\mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_7, \mathbf{a}_9\}$.
8. if $\mathbf{c}_3 = \mathbf{a}_5$, then $\mathcal{R}(T, \mathbf{c}_3) = \{\mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_{10}, \mathbf{a}_{12}\}$, so $\mathbf{c}_4 \in \mathcal{A}_3 = \{\mathbf{a}_7, \mathbf{a}_9\}$. Therefore, we get $[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_7], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_9]$.
9. if $\mathbf{c}_3 = \mathbf{a}_7$ or \mathbf{a}_8 , then similar to step 5, 6, $\mathbf{c}_4 = \mathbf{a}_8$ or \mathbf{a}_9 respectively. Then we get $[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_7, \mathbf{a}_8], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_9]$.
10. if $\mathbf{c}_2 = \mathbf{a}_5$ in step 3, then $\mathcal{R}(T, \mathbf{c}_2) = \{\mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_{10}, \mathbf{a}_{12}\}$, so $\mathbf{c}_3, \mathbf{c}_4 \in \mathcal{A}_2 = \{\mathbf{a}_7, \mathbf{a}_8, \mathbf{a}_9\}$, then similar to step 9, we get $[\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_7, \mathbf{a}_8], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_8, \mathbf{a}_9]$.
11. if $\mathbf{c}_2 = \mathbf{a}_7$ in step 3, since at this time $\#\{\mathbf{a}_3^2, \mathbf{a}_5\} = 3 > 2$, so this is impossible.
12. if $\mathbf{c}_1 = \mathbf{a}_3$ in step 2, then $\#\{\mathbf{a}_3^2\} = 2 > 1$, so this is impossible. In a word, we get

$$\begin{aligned}
& [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_3, \mathbf{a}_7], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_3, \mathbf{a}_9], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_7, \mathbf{a}_8], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_8, \mathbf{a}_9], \\
& [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_7], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_5, \mathbf{a}_9], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_7, \mathbf{a}_8], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_8, \mathbf{a}_9], \\
& [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_7, \mathbf{a}_9], [\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_5, \mathbf{a}_8, \mathbf{a}_9].
\end{aligned}$$

These form the first column of terms in $\mathcal{C}(T)$.

Remark 9. In the above example, in step 8, we can not choose $\mathbf{c}_3 = \mathbf{a}_3$ anymore, because this will be the same as step 4, and unnecessary.

Note that algorithm 3 also works for column order [4], while not holds for row orders. In the next section, we will compare the already existed straightening algorithms with our straightening algorithm.

5 Comparison of the efficiency of these straightening algorithms from examples

In this section, we will compare the four straightening algorithms: van der Waerden straightening algorithm (**vw**), N. White's second implementation algorithm of the classical straightening algorithm (**white**), Rota's straightening algorithm (**rota**) and column bracket straightening algorithm (**cb**), with specific examples.

In **rota**, we take the number of nonzero elements of the coefficient matrix plus the costs of computing all straight tableaux as the criterion. In **vw**, **white** and **cb**, we use the number of terms after combination in each step, the number of steps and the number

of all terms as criterions. For example, $[14, 7, 2], 3, 14, 23$ means that there are 3 steps of this straightening procedure, there are 14, 7, 2 terms in each step respectively, the maximal step is 14, and all the terms are 23.

1. 3×3 case:

$$\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_8 & \mathbf{a}_9 \\ \mathbf{a}_2 & \mathbf{a}_6 & \mathbf{a}_7 \\ \mathbf{a}_3 & \mathbf{a}_4 & \mathbf{a}_5 \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_6 & \mathbf{a}_9 \\ \mathbf{a}_2 & \mathbf{a}_3 & \mathbf{a}_7 \\ \mathbf{a}_4 & \mathbf{a}_5 & \mathbf{a}_8 \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_5 & \mathbf{a}_7 \\ \mathbf{a}_2 & \mathbf{a}_6 & \mathbf{a}_8 \\ \mathbf{a}_3 & \mathbf{a}_4 & \mathbf{a}_9 \end{bmatrix}$$

vw:

- 1 $[5, 3, 7, 11, 15, 19, 23, 27, 31, 32, 30, 32, 36, 40, 41, 45, 47, 49, 46, 50, 54, 53, 57, 56, 56, 53, 55, 54, 51, 51, 47, 44, 45, 41, 41, 41, 39, 38, 35, 34, 32, 30, 28, 25, 23, 19, 19, 18, 18, 15, 11, 11, 10, 10, 7, 3], 56, 56, 1813;$
- 2 $[5, 3, 7, 5, 5, 7, 7, 11, 9, 9, 10], 11, 11, 78;$
- 3 $[5, 9, 13, 17, 21, 25, 23, 27, 31, 32, 36, 34, 32, 30, 32, 32, 29, 28, 30, 29, 29, 26, 23, 23, 23], 25, 36, 639.$

white:

- 1 $[3, 5, 7, 9, 11, 12, 14, 17, 13, 9], 10, 17, 100;$
- 2 $[3, 5, 6, 9, 10], 5, 10, 33;$
- 3 $[4, 8, 11, 14, 17, 19, 22, 26, 30, 28, 32, 29, 29, 30, 30, 27, 27, 23, 24, 24, 25], 21, 32, 479.$

rota: 391.

cb:

- 1 $[14, 7, 2], 3, 14, 23;$
- 2 $[12, 13, 9, 8, 10, 11, 5, 4, 3, 2], 10, 13, 77;$
- 3 $[13, 15, 18, 19, 19, 17, 18, 17, 16, 15, 18, 13, 13, 11, 10, 9, 8, 7, 7, 5, 4, 3, 2], 23, 19, 277.$

2. 4×3 case:

$$\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_{11} & \mathbf{a}_{12} \\ \mathbf{a}_2 & \mathbf{a}_9 & \mathbf{a}_{10} \\ \mathbf{a}_3 & \mathbf{a}_7 & \mathbf{a}_8 \\ \mathbf{a}_4 & \mathbf{a}_5 & \mathbf{a}_6 \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_8 & \mathbf{a}_{12} \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_{11} \\ \mathbf{a}_3 & \mathbf{a}_6 & \mathbf{a}_{10} \\ \mathbf{a}_4 & \mathbf{a}_5 & \mathbf{a}_9 \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_{10} & \mathbf{a}_{11} \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_{12} \\ \mathbf{a}_3 & \mathbf{a}_6 & \mathbf{a}_8 \\ \mathbf{a}_4 & \mathbf{a}_5 & \mathbf{a}_9 \end{bmatrix}.$$

vw: no results (**vw** didn't get the results within 2 hours).

white:

- 1 $[3, 5, 7, 9, 11, \dots, 87, 88, 89, 85, 81], 135, 143, 12459;$
- 2 $[3, 5, 7, 9, 11, \dots, 120, 121, 121, 121], 296, 186, 36849;$
- 3 $[4, 7, 9, 11, 13, \dots, 181, 182, 183, 181], 258, 220, 42641.$

rota: 5502.

cb:

- 1 $[54, 65, 76, 78, \dots, 4, 3, 2], 77, 76, 3266;$
- 2 $[45, 54, 57, 49, \dots, 6, 6, 4, 3, 2], 121, 60, 4656;$
- 3 $[55, 64, 73, 79, \dots, 5, 4, 3, 2], 177, 119, 12564.$

3. 5×3 case:

$$\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_{14} & \mathbf{a}_{15} \\ \mathbf{a}_2 & \mathbf{a}_{12} & \mathbf{a}_{13} \\ \mathbf{a}_3 & \mathbf{a}_{10} & \mathbf{a}_{11} \\ \mathbf{a}_4 & \mathbf{a}_8 & \mathbf{a}_9 \\ \mathbf{a}_5 & \mathbf{a}_6 & \mathbf{a}_7 \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_3 & \mathbf{a}_9 \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_8 \\ \mathbf{a}_4 & \mathbf{a}_{10} & \mathbf{a}_{15} \\ \mathbf{a}_5 & \mathbf{a}_{11} & \mathbf{a}_{13} \\ \mathbf{a}_6 & \mathbf{a}_{12} & \mathbf{a}_{14} \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_6 & \mathbf{a}_{10} \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_{11} \\ \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_{12} \\ \mathbf{a}_4 & \mathbf{a}_9 & \mathbf{a}_{15} \\ \mathbf{a}_5 & \mathbf{a}_{13} & \mathbf{a}_{14} \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_6 & \mathbf{a}_{11} \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_{12} \\ \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_{13} \\ \mathbf{a}_4 & \mathbf{a}_9 & \mathbf{a}_{15} \\ \mathbf{a}_5 & \mathbf{a}_{10} & \mathbf{a}_{14} \end{bmatrix},$$

$$\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_{10} & \mathbf{a}_{15} \\ \mathbf{a}_2 & \mathbf{a}_9 & \mathbf{a}_{14} \\ \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_{13} \\ \mathbf{a}_4 & \mathbf{a}_7 & \mathbf{a}_{12} \\ \mathbf{a}_5 & \mathbf{a}_6 & \mathbf{a}_{11} \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_6 & \mathbf{a}_{10} \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_{12} \\ \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_{15} \\ \mathbf{a}_4 & \mathbf{a}_9 & \mathbf{a}_{14} \\ \mathbf{a}_5 & \mathbf{a}_{11} & \mathbf{a}_{13} \end{bmatrix}, \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_6 & \mathbf{a}_{13} \\ \mathbf{a}_2 & \mathbf{a}_7 & \mathbf{a}_{15} \\ \mathbf{a}_3 & \mathbf{a}_8 & \mathbf{a}_{12} \\ \mathbf{a}_4 & \mathbf{a}_9 & \mathbf{a}_{14} \\ \mathbf{a}_5 & \mathbf{a}_{10} & \mathbf{a}_{11} \end{bmatrix}.$$

The data of all terms are

white: 1324824 171310 14661 1415104 24345233 1321347 35537137
cb: 29453 35892 4272 116684 856398 59161 634952

The data of maximal term are

white: 1069 422 143 918 3090 846 4083
cb: 237 195 61 325 797 240 762

The data of all steps are

white: 2052 571 155 2090 11020 2161 11581
cb: 267 334 112 589 1719 423 1396

4. 6×3 case:

$$\begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_{17} & \mathbf{a}_{18} \\ \mathbf{a}_2 & \mathbf{a}_{15} & \mathbf{a}_{16} \\ \mathbf{a}_3 & \mathbf{a}_{13} & \mathbf{a}_{14} \\ \mathbf{a}_4 & \mathbf{a}_{11} & \mathbf{a}_{12} \\ \mathbf{a}_5 & \mathbf{a}_9 & \mathbf{a}_{10} \\ \mathbf{a}_6 & \mathbf{a}_7 & \mathbf{a}_8 \end{bmatrix}.$$

The data of [all terms, maximal term, all steps] are

white: no results after 24 hours
cb: [1827595, 1624, 2037]

5. The Turnbull-Young invariant polynomial [15]. This polynomial describes the condition of 10 points lie on a quadric surface in 3-dimensional projective space.

$$H = \{(\mathbf{a}_2\mathbf{a}_3\mathbf{a}_4\mathbf{a}_5\mathbf{a}_6)\}\{\mathbf{a}_7\mathbf{a}_8\mathbf{a}_9\}'[\mathbf{a}_0\mathbf{a}_1\mathbf{a}_2\mathbf{a}_3][\mathbf{a}_0\mathbf{a}_4\mathbf{a}_5\mathbf{a}_6][\mathbf{a}_1\mathbf{a}_5\mathbf{a}_7\mathbf{a}_8][\mathbf{a}_2\mathbf{a}_6\mathbf{a}_8\mathbf{a}_9][\mathbf{a}_3\mathbf{a}_4\mathbf{a}_7\mathbf{a}_9],$$

$$K = \{1 - (\mathbf{a}_0\mathbf{a}_7) - (\mathbf{a}_1\mathbf{a}_7) - (\mathbf{a}_2\mathbf{a}_7) - (\mathbf{a}_3\mathbf{a}_7) - (\mathbf{a}_4\mathbf{a}_7) - (\mathbf{a}_5\mathbf{a}_7) - (\mathbf{a}_6\mathbf{a}_7)\}H,$$

where $\{(\mathbf{a}_2\mathbf{a}_3\mathbf{a}_4\mathbf{a}_5\mathbf{a}_6)\}$ denotes the five-term cyclic permutation group, $\{\mathbf{a}_7\mathbf{a}_8\mathbf{a}_9\}'$ denotes the group of all permutation of $\{\mathbf{a}_7, \mathbf{a}_8, \mathbf{a}_9\}$, and $(\mathbf{a}_i\mathbf{a}_7)$ refers to a permutation of \mathbf{a}_i and \mathbf{a}_7 , $i = 0, \dots, 6$. Thus K has 240 terms. Finally, the Turnbull-Young invariant polynomial equals to $-\frac{1}{20}K$. Here, we only straight K with the method of **white**

and **cb**. This has already been achieved in [17]. Our result is that the straightening expression of K has 473 terms, and the corresponding data of us are:

method	steps	max term	all terms
vw :	20834	6457	1.0×10^9
white :	9845	2841	1.9×10^8
cb :	713	678	3.8×10^5

From the above examples, we see that in the 3×3 case, **white** is a much better improvement of **vw**. By contrast, **white** and **cb** are equally matched. In the $4 \times 3, 5 \times 3, 6 \times 3, 5 \times 4$ case, **cb** reflects more advantages than **white**. Since **rota** is a fixed procedure, so it's hard to compare it with others.

6 Conclusion

In this paper, we present another straightening algorithm in bracket algebra. The examples shown in section 5 tell us that this new straightening algorithm is much better than the others. Our algorithm mainly based on the concept column brackets. Column brackets have a very strong symmetry, so its hard to find some other new results or applications. For example, column bracket do not have the concept the normal forms, and so do not have straightening algorithms. Thus we can not change a column bracket monomial into a column bracket polynomial with lower orders. But column bracket is better than Rota's straightening algorithm in explanting straightening coefficients (see Appendix 2).

References

1. Barnabei, M., Brini, A. and Rota, G.-C. On the Exterior Calculus of Invariant Theory. *J. of Algebra* **96**: 120-160, 1985.
2. Clausen, M. Multivariate Polynomials, Standard Tableaux, and Representations of Symmetric Groups. *J. Symbolic Computation* **11**, 483-522, 1991.
3. Clausen, M. Dominance Orders, Capelli Operators, and Straightening of Bideterminants, *Europ. J. Combinatorics* **5**, 207-222, 1984.
4. Désarménien, J., Kung, J. P. S. and Rota, G.-C. Invariant Theory, Young Bitableaux, and Combinatorics. *Advances in Mathematics* **27**, 63-92, 1978.
5. Désarménien, J. An algorithm for the Rota Straightening Formula. *Discrete Mathematics* **30**, 51-68, 1980.
6. Doubilet, P., Rota, G.-C. and Stein, J. On the Foundations of Combinatorial Theory: IX Combinatorial Methods in Invariant Theory. *Stud. Applied Math* **53**, 185-216, 1974.
7. Huang, R. Q. and White, N. L. Straightening Coefficients in the Supersymmetric Letter-Place Algebra. *Journal of Algebra* **171**, 655-675, 1995.
8. Li, H. *Invariant Algebras and Geometric Reasoning*. World Scientific, Singapore, 2008.
9. Li, H., Shao C., Huang L., Liu Y. Reduction among Bracket Polynomials, *In: Proc. ISSAC'14*, 304-311, 2014.
10. McMilln, T. and White, N. The Dotted Straightening Algorithm. *J. Symbolic Computation* **11**, 471-482, 1991.

11. Minc, H. *Permanents*. Encyclopedia of Mathematics and Its Applications, Vol. **6**, Cambridge University Press, 1978.
12. Rutherford D. E. *Substitutional Analysis*. Edinburgh University Press. 1948.
13. Sagan B. E. The Ubiquitous Young Tableau. Invariant Theory and Tableaux. In: Staton, D. (ed.), *Invariant Theory and Tableaux*, Springer, New York. 262-298. 1988.
14. Sturmfels, B. *Algorithms in Invariant Theory*. Second edition, SpringerWien-NewYork, 2008.
15. Turnbull, H. W. and Young A. The Linear Invariants of Ten Quaternary Quadrics. Trans. Camb. Phil. Soc. **23**: 265-301, 1926.
16. Weyl, H. *The Classical Group-Their Invariants and Representations*. Princeton University Press. 1939.
17. White, N. Implementation of the Straightening Algorithm of Classical Invariant Theory. In: Staton, D. (ed.), *Invariant Theory and Tableaux*, Springer, New York, 36-45, 1990.
18. White N. Cayley factorization and a straightening algorithm. In *Topics in Computational Algebra*, 163C184. Springer, 1990.
19. White, N. Multilinear Cayley Factorization. *J. Symbolic Computation* **11**, 421-438, 1991.
20. Young, A. *The Collected Papers of Alfred Young, 1873-1940*. University of Toronto Press, 1977.

7 Appendix 1: Some figures

In the 3×3 and 4×3 case, we show the data all the terms, all the steps and all the terms times all the steps of the each method in the following pictures. The green one is **vw**, the red one is **white**, the black one is **rota**, the blue one is **cb**.

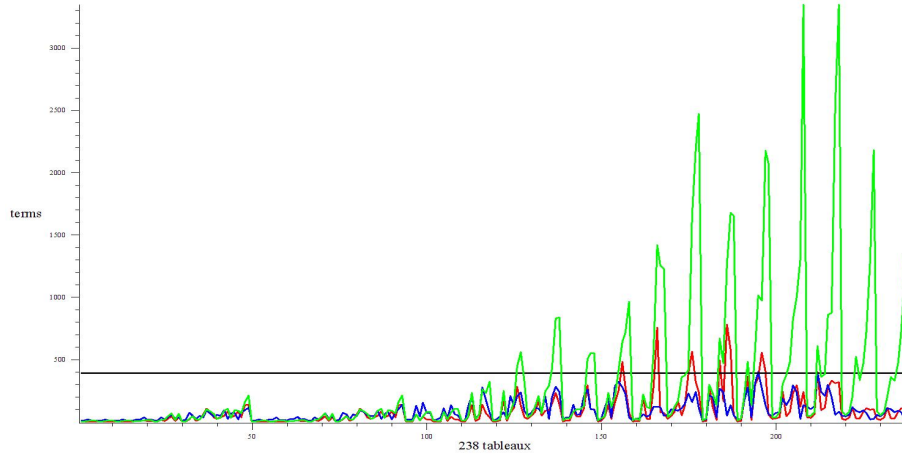


Fig. 1. graph of terms of 238 3×3 non-straight tableaux

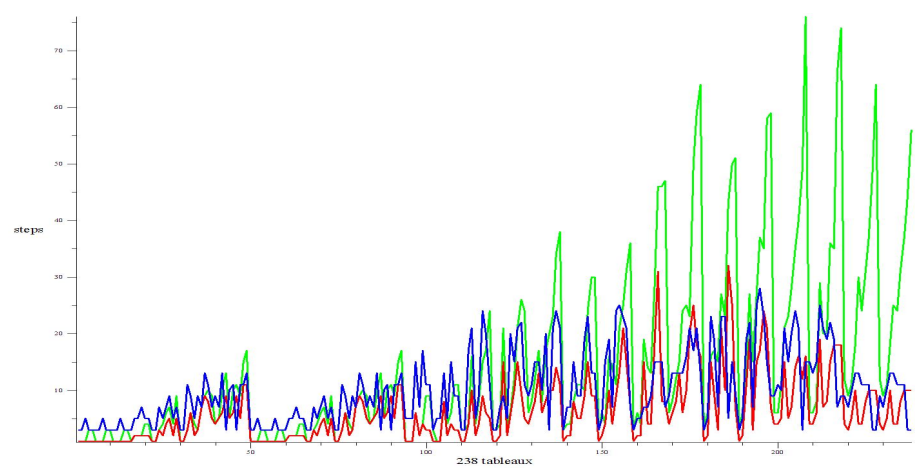


Fig. 2. graph of steps of 238 3×3 non-straight tableaux

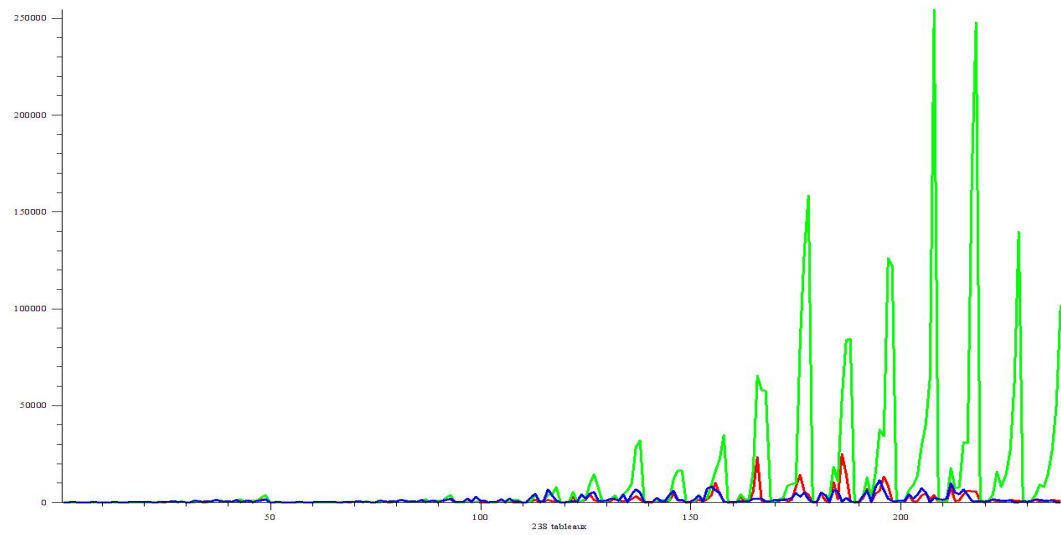


Fig. 3. steps \times terms of 238 3×3 non-straight tableaux

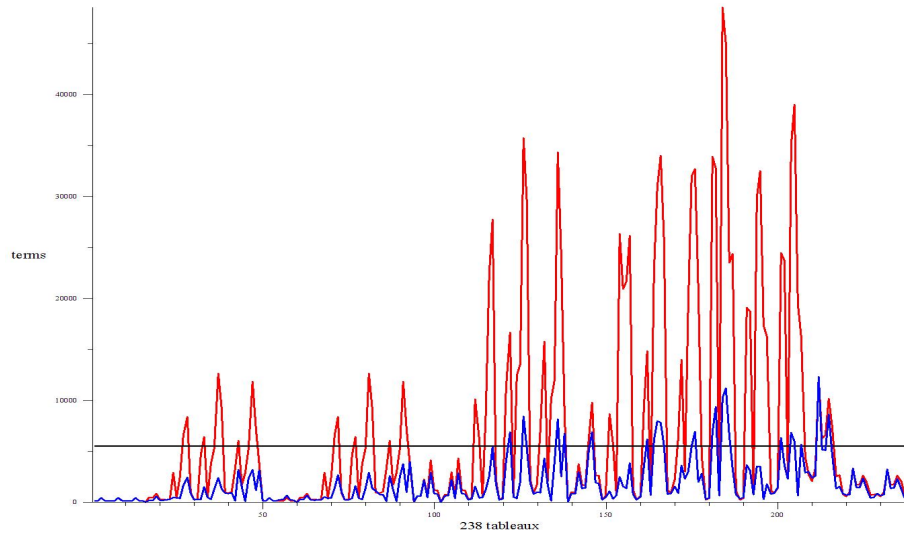


Fig. 4. graph of terms of 238 4×3 non-straight tableaux

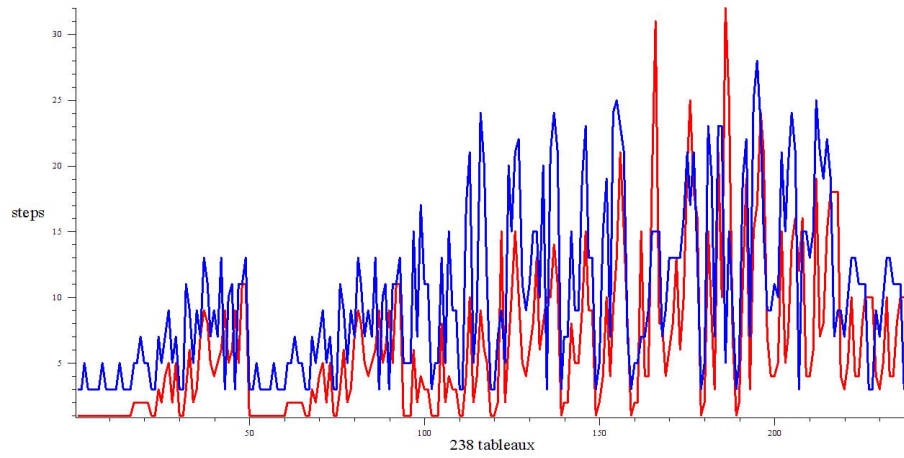


Fig. 5. graph of steps of 238 4×3 non-straight tableaux

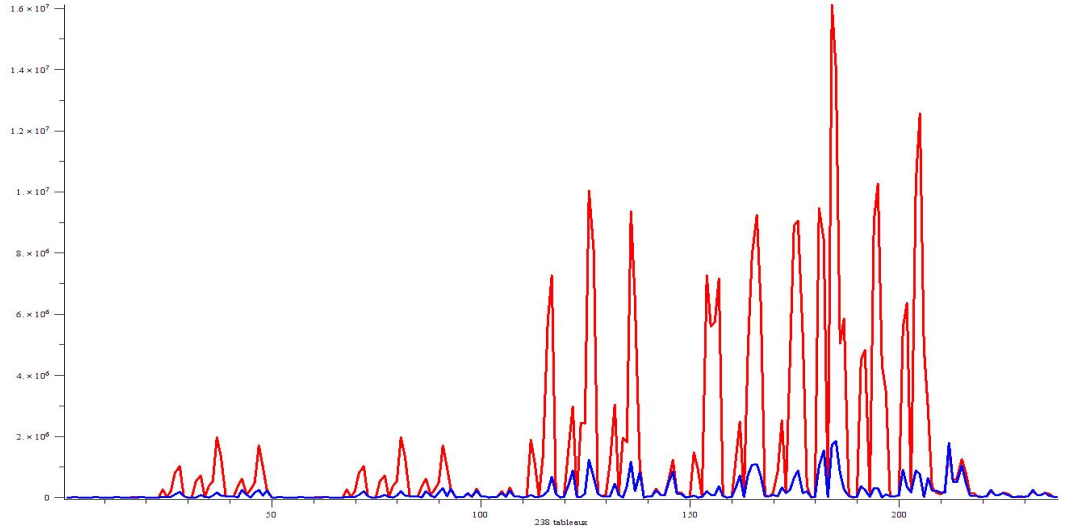


Fig. 6. steps \times terms of 238 4×3 non-straight tableaux

8 Appendix 2: Straightening coefficients

In this section, we will consider the formula of straightening coefficients. This have been studied in [3], [5], [7], etc.. Their proofs are based on Rota's straightening algorithm [4] and a technique raised by Clausen [2]. But the proof here is quite obvious, and the understanding of this formula is more clear. The following definition comes from the definition of $\mathcal{P}_c(T)$.

Definition 8. Let T_1, T_2 be two tableaux of degree d , we define

$$\xi(T_1, T_2) = \sum_X \text{rowsign}(X, T_1), \quad (8.1)$$

where

(1) the summation over all X such that each row (resp. column) of X can be transformed into T_1 (resp. T_2) by some permutations.

(2) if we denote δ_i as the permutation's sign of i -th row of X with respect to that of T_1 , then

$$\text{rowsign}(X, T_1) = \prod_{i=1}^d \delta_i. \quad (8.2)$$

Example 14. $T_1 = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_3 \mathbf{a}_5 \\ \mathbf{a}_2 \mathbf{a}_4 \mathbf{a}_6 \end{bmatrix}$, $T_2 = \begin{bmatrix} \mathbf{a}_1 \mathbf{a}_2 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_5 \mathbf{a}_6 \end{bmatrix}$, then such X can be $\begin{bmatrix} \mathbf{a}_1 \mathbf{a}_5 \mathbf{a}_3 \\ \mathbf{a}_4 \mathbf{a}_2 \mathbf{a}_6 \end{bmatrix}$, and $\text{rowsign}(X, T_1) = 1$.

The following lemma gives us a direct explanation of the definition of $\xi(T_1, T_2)$.

Lemma 1. Assume that $\mathcal{P}_c(T) = \sum \lambda_i T_i$. Then

$$\lambda_i = \xi(T, T_i). \quad (8.3)$$

According to lemma 1 and Algorithm 3, we have the following corollary about straightening coefficients which seems more obvious in the sense of column bracket.

Corollary 5. *Let T be a bracket monomial. Let $T = \sum \lambda_i T_i$ be the straight expression of T in the descending form under the negative column order, then*

$$\lambda_i = \xi(T, T_i) - \sum_{j < i} \lambda_j \xi(T_j, T_i). \quad (8.4)$$

Remark 10. From (8.4), we find that the straightening coefficient λ_i depends not only on T , but also on T_j , $j < i$. Note that $\lambda_1 = 1$.

9 Appendix 3: straightening expression of Turnbull-Young invariant polynomial

$$\begin{aligned}
& -12[0123]^2[4567][6789][4589] + 2[0246][1367][2589][0134][5789] + 8[0125][3467][5789][0123][4689] - 2[0123][0246][4789][3589][1567] \\
& + 2[0124][0236][4789][3589][1567] - 2[0246][1357][3489][6789][0125] + 2[0126][3457][3489][6789][0125] - 2[0124][3789][3689][1567][0245] \\
& + 2[0247][1467][3589][0123][5689] - 2[0134][0234][1567][2589][6789] + 2[0157][2467][0123][5689][3489] + 2[0237][1457][5689][0124][3689] \\
& - 6[1467][3489][0257][5689][0123] + 2[0124][0235][1357][4689][6789] + 2[0124][0236][1357][6789][4589] - 2[1467][2489][2589][0367][0135] \\
& - 2[0137][2457][5689][0123][4689] + 2[0136][0247][2589][6789][0345][0124] - 2[2357][2689][4789][0134][0156] \\
& - 6[0125][6789][0136][4589][2347] - 2[1267][4589][5689][0237][0134] + 2[1467][2589][0235][0134][6789] + 2[1467][2589][0237][0134][5689] \\
& + 2[1467][2589][2689][0347][0135] + 2[3489][5789][0135][0246][1267] + 2[0257][1357][3689][0124][4689] + 2[0134][5789][2689][1567][0234] \\
& - 2[0247][1367][5689][2589][0134] - 2[1467][2589][0237][0135][4689] - 2[2367][3489][0125][0147][5689] - 2[0147][2357][2689][0134][5689] \\
& + 2[2357][2689][0146][5789][0134] - 2[0124][0346][1357][6789][2589] - 4[0134][0237][4689][2589][1567] + 2[0247][1357][0134][5689][2689] \\
& - 10[0123][0246][1357][6789][4589] - 2[0134][0235][4789][2689][1567] - 2[0124][3689][6789][0245][1357] - 2[1467][3589][0267][0123][4589] \\
& + 4[0126][5789][0136][4589][2347] - 8[0123][5789][3689][1457][0246] + 2[1347][5689][0123][6789][0245] + 2[3489][6789][0125][1457][0236] \\
& - 2[0135][0246][1257][4789][3689] - 2[0127][3457][4689][0126][3589] + 2[0123][0256][1457][4789][3689] + 2[1367][5689][5789][0124][0234] \\
& + 8[0127][3457][5689][4689][0123] + 2[2367][2589][0147][5689][0134] - 6[2567][4689][0137][4589][0123] + 2[0235][1267][4789][0135][4689] \\
& + 2[1257][4689][0237][0136][4589] + 2[2467][3589][0123][0167][4589] - 2[1467][2589][0347][3589][0126] - 8[0123][0245][1346][5789][6789] \\
& - 4[1367][2489][5689][0357][0124] + 2[1267][3489][0145][0257][3689] - 2[0134][0246][1357][5789][2689] - 6[2467][3589][0123][0157][4689] \\
& - 2[2589][4789][0126][0346][1357] - 2[0234][1267][5789][5689][0134] + 2[0257][1367][2489][0135][4689] - 4[0145][2367][5789][0123][4689] \\
& + 4[0136][0124][2457][3589][6789] - 4[2457][3689][0156][4789][0123] - 2[1357][2689][4689][0125][0347] - 2[0124][5789][3689][1457][0236] \\
& + 2[0123][0256][3489][5789][1467] + 2[1267][3489][4589][0136][0257] + 2[0347][1357][2589][4689][0126] + 2[2347][5689][0125][0137][4689] \\
& + 2[3467][3589][0127][5689][0124] - 2[1347][2689][0245][6789][0135] + 2[0125][0345][3789][2689][1467] - 2[0134][0236][1257][5789][4689] \\
& - 6[0123][0457][3689][2589][1467] - 4[1267][3489][0245][0135][6789] - 2[1467][3489][0235][0125][6789] - 12[0123][2567][6789][0134][4589] \\
& - 2[0134][0235][1256][4789][6789] + 4[0123][0257][4689][3589][1467] + 2[1467][2489][0126][0357][3589] + 2[0246][1347][0126][3589][5789] \\
& + 2[0124][3467][5789][0123][5689] + 2[1347][2689][5689][0125][0347] + 2[0123][0246][1467][5789][3589] + 2[0124][3567][4789][3589][0126] \\
& + 4[0123][0146][2356][4789][5789] - 2[0124][0136][2356][4789][5789] - 2[2489][3589][0367][1467][0125] \\
& - 2[0123][0156][2367][4789][4589] + 4[2357][4689][0137][0124][5689] + 2[2467][3589][0137][0124][5689] + 2[1357][2689][4789][0125][0346] \\
& - 2[1567][3689][0124][0135][0245] + 2[2467][4589][0137][0123][5689] + 2[1357][2689][4789][0125][0346] + 2[0123][0156][2467][3589][4789] \\
& - 2[0134][0256][1367][2589][4789] + 2[0135][0267][2489][3589][1467] - 4[0126][3567][4589][0123][4789] - 4[0146][2567][0123][4789][3589] \\
& - 2[0124][0345][1367][2689][5789] - 6[0124][0126][3457][3589][6789] + 4[2567][4589][0136][4789][0123] - 6[1357][4689][0247][5689][0123] \\
& + 18[0123][0124][3567][4589][6789] + 4[0123][0234][1456][5789][6789] - 2[0123][0256][1357][4689][4789] - 4[0124][3589][2467][0135][6789] \\
& - 2[2567][3689][0124][0135][4789] - 2[0124][3689][5789][1367][0245] + 2[0126][0137][2457][3589][4689] - 2[0356][1457][2789][2689][0134] \\
& - 2[1467][2489][0135][0257][3689] - 2[0256][1357][4789][0124][3689] + 6[1267][3489][0134][0257][5689] - 2[1356][4789][0124][0235][6789] \\
& - 4[0236][1247][5789][4589][0136] - 2[1267][4589][0236][0135][4789] + 6[0123][2467][6789][4589][0135] + 6[0123][0156][2357][4789][4689] \\
& + 2[0124][0347][3689][2589][1567] + 2[0237][1357][4689][0124][5689] + 4[0246][1357][3589][0124][6789] - 4[2347][4589][0126][0137][5689] \\
& + 4[0123][3457][0126][4689][5789] + 2[0236][1457][4789][0126][3589] - 6[1257][3489][0136][5789][0246] + 2[2467][2589][0136][5789][0134] \\
& - 2[0135][0247][1457][2689][3689] - 4[1267][3489][5789][0134][0256] + 2[1367][2689][0125][0345][4789] + 4[1367][2589][0124][0367][4589] \\
& + 2[1567][4689][0235][0123][4789] + 2[1367][3489][0245][6789][0125] + 2[0236][1257][4689][4789][0135] + 2[0357][1467][2489][0135][2689] \\
& - 2[0136][4789][2589][0134][2567] - 8[0123][4689][5789][1567][0234] - 2[1357][2589][6789][0246][0134] + 2[2489][6789][0125][0345][1367] \\
& + 4[1457][2689][0123][0345][6789] + 2[0124][2367][5789][3689][0145] - 2[0123][2467][5789][3689][0145] + 2[0134][2567][0124][6789][3589] \\
& + 2[0237][1257][4689][0134][5689] + 2[0136][0246][1257][4789][3589] + 4[0125][0135][2347][4689][6789] - 2[0124][5789][3689][1567][0234] \\
& - 2[0125][0346][1347][2689][5789] + 4[0236][1257][4589][0134][6789] - 2[0357][1367][2489][4589][0126] + 2[0135][4789][2689][0134][2567] \\
& - 2[0134][0267][1467][3589][2589] + 2[0123][4689][5789][1467][0235] - 4[0136][0247][1247][5689][3589] - 2[3589][6789][0146][2357][0124] \\
& - 14[0123][3457][0126][4689][5789] + 2[0236][1457][4789][0126][3589] - 2[0236][1257][4589][4789][0136] - 8[0123][0345][1467][2589][6789] \\
& - 2[2367][4589][0136][0125][4789] + 2[0123][0256][1367][4789][4589] + 4[0136][0246][1247][5789][3589] - 6[0124][2357][5789][0136][4689] \\
& + 6[0123][2457][5789][0136][4689] - 4[0123][4789][4689][0135][2567] + 2[1346][5789][6789][0124][0235] + 4[0123][0156][2357][4789][4689] \\
& - 2[2457][4689][6789][0123][0135] + 2[0124][0256][1457][3789][3689] - 2[0124][0134][2356][5789][6789] + 4[2567][3589][4689][0137][0124] \\
& - 2[0246][1357][3589][4789][0126] + 2[0134][2367][0145][2589][6789] + 2[1567][2589][0346][4789][0123] - 2[0237][1467][5689][0124][3589] \\
& + 2[0136][2357][4789][0126][4589] - 2[0145][2367][6789][0125][3489] + 2[0146][2357][6789][0125][3489] + 10[0236][1467][4589][5789][0123] \\
& + 2[2467][3489][0156][0123][5789] - 2[1467][2589][0256][3789][0134] - 4[2456][3789][0135][6789][0124] - 2[0134][0235][1247][6789][5689] \\
& - 10[0123][0246][1367][4589][5789] + 2[0246][1357][0124][3689][5789] + 6[3489][5689][0247][1257][0136] - 2[0124][0356][1367][2589][4789] \\
& + 2[0137][2457][5689][0134][2689] - 2[0124][0135][2467][3689][5789] - 2[2467][3589][0124][5789][0136] + 4[0134]^2[2567][6789][2589] \\
& - 6[0124]^2[3567][3589][6789] + 2[0126]^2[3457][3589][4789] - 2[0125]^2[3467][3489][6789] - 2[1257][3489][0146][0257][3689] \\
& - 8[1467][4589][0123][0235][6789] - 6[0347][1357][2489][5689][0126] - 2[1267][3589][0145][0247][3689] - 4[0257][1367][4689][4589][0123] \\
& - 4[0257][1367][3589][0124][4689] - 2[0124][0137][2457][3689][5689] - 2[0237][1457][4689][3589][0126] + 2[0123][2457][5789][3689][0146] \\
& - 2[0247][1257][0134][3689][5689] + 4[2589][4689][0137][2567][0134] - 2[1457][2589][2689][0347][0136] - 2[2589][4689][0123][0347][1567] \\
& - 2[0125][0346][3789][2689][1457] + 4[0124][0357][1367][4689][2589] - 6[0123][0456][3789][2689][1457] + 2[0247][1367][2589][0135][4689] \\
& - 2[0134][5789][2689][1467][0235] + 2[0135][4789][2689][1467][0235] + 6[0135][0236][1247][4589][6789] - 2[0124][0357][1357][4689][2689] \\
& + 2[0247][1267][5689][0134][3589] + 6[0123][0457][1457][2689][3689] - 2[0125][0347][1367][4689][2589] + 8[2347][5689][0124][0136][5789] \\
& - 2[0136][0134][2457][2689][5789] - 2[1457][3689][0257][0123][4689] + 2[2489][6789][0135][1457][0236] - 2[2689][4789][0135][1457][0236] \\
& + 8[0123][4589][6789][0245][1367] - 2[2367][2589][0146][5789][0134] - 2[2456][3789][0123][5789][0146] + 10[0147][2567][4689][0123][3589] \\
& + 6[0123][2457][6789][0145][3689] + 4[0346][1456][0123][2789][5789] - 4[1267][3589][4789][0134][0256] - 4[0356][1467][2467][2489][5789][0123] \\
& - 4[0357][1467][2689][2589][0134] - 4[0123][2456][5789][4789][0136] - 2[1367][2589][0347][0124][5689] + 2[1357][2689][0245][4689][0134] \\
& + 8[2457][3489][0123][0156][6789] + 8[0123][3467][4589][5789][0126] + 2[0125][2467][4789][3589][0136] + 2[1467][2589][5689][0347][0126] \\
& + 4[1367][2589][5789][0124][0346] + 2[0134][0156][2367][2589][4789] - 2[1257][3689][4789][0134][0256] + 4[0237][1247][5689][0136][4589] \\
& + 2[0345][1347][0125][6789][2689] - 4[0357][1457][0123][4689][2689] + 4[0123][0246][1456][5789][3789] + 2[0246][1357][0124][3689][5789] + 2[0246][1357][2589][0136][4789] \\
& - 2[3489][6789][0134][0256][1257] + 2[0135][2689][3789][1457][0246] - 2[2467][4589][0136][0123][5789] - 6[0135][0246][1247][6789][3589] \\
& + 6[0124][0135][2367][4689][5789] + 4[1367][2489][0347][5689][0125] - 2[0126][0247][3589][3689][1457] - 2[1367][4589][0247][3589][0126]
\end{aligned}$$

$$\begin{aligned}
& +2[0234][1357][0124][5689][6789] +2[0247][1367][5689][0124][3589] +8[0145][2347][0123][5689][6789] +2[0134][0157][2357][4689][2689] \\
& +2[2489][5789][0246][1367][0135] +4[0346][1457][0123][6789][2589] +4[0247][1457][0123][3689][5689] +2[1267][3489][4589][0267][0135] \\
& +4[1467][3589][0235][0124][6789] -4[0346][1347][2589][0126][5789] +4[0123][0345][1467][2689][5789] -2[1247][3689][0135][0247][5689] \\
& -2[1457][2689][0237][0134][5689] +2[0347][1357][0124][5689][2689] -6[0145][2367][6789][0123][4589] -2[0247][1357][0124][5689][3689] \\
& -2[2467][3589][0137][4589][0126] -2[2357][2689][0145][6789][0134] -2[0145][2346][0123][5789][6789] -2[1357][2589][0136][0247][4689] \\
& -2[2357][4689][0126][0137][4589] +2[0124][2347][6789][0135][5689] +2[0125][2467][6789][3489][0135] +2[2367][5689][0134][0157][2489] \\
& -4[0247][1567][3689][0124][3589] -2[0357][1367][2489][2689][0145] +6[0123][0146][2367][4589][5789] -2[0124][2357][6789][0136][4589] \\
& +6[0123][0145][2356][6789][4789] -2[0125][0136][2457][3489][6789] +4[0347][1457][0123][5689][2689] -2[1367][2489][0135][0267][4589] \\
& -2[0247][1347][5689][3589][0126] +4[0123][0357][1467][2589][4689] -2[0236][1247][4689][5789][0135] +2[0237][1467][4689][0125][3589] \\
& +2[1257][3689][4689][0257][0134] -6[0123][0147][2467][3589][5689] -4[0123][2457][5689][0134][6789] +2[0236][1467][2589][0134][5789] \\
& +4[0124][5789][3689][0134][2567] -2[0124][0357][1467][2589][3689] -2[0245][1367][5789][0123][4689] +2[0124][0356][1357][2689][4789] \\
& -2[0246][1347][2589][0136][5789] -8[0123][0246][1356][4789][5789] -2[0136][0124][2367][4589][5789] +2[0135][0246][3789][2589][1467] \\
& -2[0235][1347][6789][4689][0125] -2[0237][1347][4689][0125][5689] +2[1367][4589][5689][0124][0237] +4[0246][1347][2589][6789][0135] \\
& +4[0356][1457][0123][2689][4789] -2[0127][3467][0125][5689][3489] +2[0137][2467][0125][5689][3489] +4[0123][2467][3589][3489][1467][0245] \\
& +14[4689][5789][0123][0246][1357] +6[0123][0246][1457][3589][6789] -2[0245][1267][0134][3589][6789] +2[3467][3589][0127][0126][4589] \\
& -2[0125][3467][3689][0124][5789] -2[1567][2689][0123][0345][4789] -2[1457][2689][1357][4689][5789] +2[0237][1457][4689][0135][2689] +2[0346][1457][2589][2789][0136] \\
& +4[0125][0346][1347][6789][2589] +2[0123][0147][2367][4589][5689] +2[0124][0235][1567][3689][4789] -4[0146][2346][0123][5789]^2 \\
& -2[0124][0157][2367][3489][5689] +4[0124][3456][6789][0125][3789] +6[0123][0145][4789][3689][2567] \\
& -2[489][5789][0236][1467][0135] -6[0123][2467][6789][0145][3589] -2[1347][3589][0246][6789][0125] +8[0123][0146][2357][4589][6789] \\
& +4[0123][0134][2567][4689][5789] +2[2457][2689][0135][6789][0134] -2[0257][1467][0126][3589][3489] +8[0125][3467][0124][6789][3589] \\
& +2[0134][0167][4589][2589][2367] +2[0267][1467][0125][3589][3489] +4[1267][3589][4589][0134][0267] -4[1457][4689][0236][0123][5789] \\
& -10[0123][3467][0125][4589][6789] +2[1456][2789][6789][0235][0134] -2[0257][1457][3689]^2[0124] +2[0346][1457][2589][2789] +2[0256][1357][0134][2689][4789] \\
& -2[0123][0256][4789][3589][1467] +2[0257][1457][3689]^2[0124] +2[0135][2345][6789]^2[0124] +4[1467][2589][0123][0346][5789] +2[0134][0356][1467][2589][2789] \\
& -2[0126][2457][3589][4789][0136] +2[0246][1367][3589][4789][0125] +4[1467][2589][0123][0346][5789] +2[0134][2357][6789][4689][0135] \\
& -2[0135][0346][1467][2589][2789] -6[0347][1467][2589][0123][5689] -6[0124][2357][6789][4689][0135] -2[0125][2347][5789][0136][4689] \\
& -8[0346][1457][0123][2689][5789] +4[0134][0257][1367][4689][2589] +2[0247][1257][3689][0146][3589] -2[0125][0236][4789][5689][1467] \\
& +6[0123][3457][6789][4589][0126] +2[0125][0236][1347][4689][5789] -2[0135][2689][3789][0245][1467] -4[0137][2467][5689][2589][0134] \\
& +2[3457][4689][0125][0123][6789] +2[1347][2589][0136][0247][5689] +2[0125][0247][3489][5689][1367] +2[0136][0134][2457][2589][6789] \\
& +4[0236][1456][5789][0123][4789] +6[0126][3456][5789][0123][4789] -4[0134][0157][2367][4689][2589] +2[0134][0246][1257][5789][3689] \\
& +2[0134][0123][2456][5789][6789] -2[1467][2589][4789][0123][0356] -2[0135][0245][1267][4789][3689] -6[0134][5789][0124][5689][2367] \\
& -8[0256][1457][6789][0123][3489] +22[3456][6789][0124][0123][5789] -10[0237][1467][5689][0123][4589] -2[0136][0247][1257][4689][3589] \\
& +2[0124][2356][4789][0135][6789] +4[0123][0157][2457][3689][4689] -2[1267][3489][0135][0247][5689] +4[2489][5689][0135][0237][1467] \\
& -2[0356][1457][2489][6789][0123] -2[2367][4589]^2[0167][0123] +4[0124][0257][3689][3589][1467] -2[1367][2589][4589][0134][0267] \\
& +2[0124][0235][1456][6789][3789] +2[1246][5789][0134][0235][6789] +2[0135][0246][1267][4789][3589] -4[0124][3457][6789][0125][3689] \\
& +6[0346][1357][2489][5789][0126] -4[1457][6789][0123][0234][5689] -4[1267][3589][0134][0257][4689] +2[1457][2689][0126][0347][4589] \\
& -10[0124][0123][3567][4689][5789] +2[1567][2689][2789][0345][0134] -2[0136][2589][3789][1457][0246] +2[1367][2589][0126][0347][4589] \\
& +6[1367][4589][0247][0123][5689] +2[1247][4689][0237][0135][5689] -4[1357][2489][6789][0125][0346] -2[3567][3689][0124]^2[5789] \\
& +4[0134][0245][1267][5789][3689] +4[0123][0147][2347][5689]^2 -2[0346][1357][0124][2689][5789] -2[2689][5789][0134]^2[2567] \\
& +2[2589][3689][0257][1467][0134] -2[2589][4789][0246][1367][0135] +2[0136][0124][2457][3689][5789] -4[1457][2589][0134][0236][6789] \\
& -2[0267][1467][3589]^2[0124] -2[0246][1367][0124][3589][5789] +2[0236][1457][2689][0134][5789] +2[0135][0246][1247][5789][3689] \\
& -2[0134][0257][1357][4689][2689] -12[0123][0246][1347][5789][5689] -4[0125][0346][1367][2489][5789] -4[2489][5689][0247][1367][0135] \\
& -2[0125][0237][3489][5689][1467] +6[2345][6789]^2[0145][0123] +2[0135][0247][1257][4689][3689] -2[1347][5689][6789][0124][0235] \\
& -2[0237][1257][4689]^2[0135] -6[0123][3457][0124][5689][6789] +12[0123]^2[4689][5789][4567] +2[1467][2589]^2[0367][0134] \\
& +2[0247][1467][3589]^2[0126] -4[0123][0157][2357][4689]^2 +2[0267][1367][4589][0124][3589] -4[2589][6789][0134][0135][2467] \\
& -2[1367][2489][0257][0134][5689] +2[1457][2689][0347][0125][3689] -4[2456][3789][0123][0145][6789] +2[1367][4589]^2[0267][0123] \\
& +2[0247][1357][4689][0126][3589] +4[0124][0356][1367][2489][5789] +2[0134][0236][1247][5789][5689] +2[0234][1257][6789][0134][5689] \\
& +8[0123][0234][1567][4589][6789] -2[1345][6789]^2[0124][0235] -6[0124][0137][2347][5689]^2 +2[0124][0237][4689][3589][1567] \\
& +2[1357][2489][4689][0126][0357] +6[0123][0157][2367][4589][4689] +2[1357][4689]^2[0257][0123] +2[2367][4589]^2[0137][0126] \\
& +2[1367][2489][2589][0145][0367] -2[2346][5789]^2[0136][0124] -2[0134][0237][1247][5689]^2 +4[0123][0456][3789][2589][1467] \\
& -2[0124][0235][1367][4689][5789] -2[0134][0235][1245][6789]^2 +2[1467][2589][0124][0367][3589] +8[0123][0246][1346][5789]^2
\end{aligned}$$